NEWSCLAIMS: A New Benchmark for Claim Detection from News with Attribute Knowledge

A Project Report Submitted

in Partial Fulfilment of the Requirements

for course work CS570

(FUNDAMENTALS OF INFORMATION RETRIEVAL)

by

GYAN RATNA-224156016
AJAY KUMAR-190101006

To

Dr. Sanasam Ranbir Singh

(Associate Professor)

Department of Computer Science and Engineering

INDIAN INSTITUTE OF TECHNOLOGY

GUWAHATI

[May 13, 2023]

Contents

Chapter 1

Introduction

The internet era has ushered in an explosion of online content creation, resulting in increased concerns regarding misinformation in news, online debates, and social media. A key element of identifying misinformation is detecting the claims and the arguments that have been presented. In this regard, news articles are particularly interesting as they contain claims in various formats: from arguments by journalists to reported statements by prominent public figures.

Claim detection and verification are crucial for news understanding and have emerged as promising technologies for mitigating misinformation and disinformation in the news. However, most existing work has focused on claim sentence analysis while overlooking additional crucial attributes (e.g., the claimer and the main object associated with the claim). In this work, we present NEWSCLAIMS, a new benchmark for attribute-aware claim detection in the news domain.

The claim detection problem to include extraction of additional attributes related to each claim and release 889 claims annotated over 143 news articles. NEWSCLAIMS1 aims to benchmark claim detection systems in emerging scenarios, comprising unseen topics with little or no training data.

Chapter 2
Proposed Claim Detection Task

Task is to detect claims related to topic and their corresponding attributes.

These are the corresponding attributes:

- **Claim Sentence Detection:**
- It is to extract sentences that are relevant to defined topics.
- It first identify sentences that contain verifiable claims according to check-worthiness.
- Then select which are related to topic. Some topics:
- Origin of virus
- Transmission of virus

- **Claimer Detection:**
- Claims can be claimed by various sources: person, report etc.
- If claim is asserted by article author then journalist is claimer.
- It involves identifying whether claim is by journalist or reported in news article.
- **Claim Object Detection:**
- It relates with what is being claimed.
- Ex. virus origin species or who created virus.
- **Stance Detection:**
- It outputs whether the task is asserting or refuting.
- It differs from task formulation used in other stance detection datasets.

- **Claim Span Detection:**
- It identifies the claim boundaries within sentence, including actual claim content**.**
-

Chapter3
Data Collection and Preprocessing

Data is the prerequisite to train any machine learning model to solve any task. Herethe task is to predict the quality of the paper just seeing the abstract. And to traina particular machine learning model for this task, data set consists of abstract along with some quality tag is required.

2112.08544.pdf (arxiv.org) Using this link of the assigned research paper task of NEWSCLAIMS: A New Benchmark for Claim Detection from News with Attribute Knowledge
The github link GitHub - blender-nlp/NewsClaims
dataset has been used for annotation and preprocessing of the datasets
The codes for training the models were not available

Data Preprocessing and Annotation

- For evaluation of performance of models on different components of our claim detection task.
- We are releasing set base on COVID-19.
  Annotation:
  It is divide in two phases:
  Identifying claim sentences with their corresponding topics.
  Annotating the attribute of these claims.

In phase 1 target sentence is displayed with red highlighted in sentence.
Then it is checked if highlighted text belong to previous defined topics.
It took ~30 sec per sentence.
In phase 2 claim sentence is displayed with red highlighted.
Then annotator is asked to identify other attributes.
It took ~90 sec to annotate the attributes.

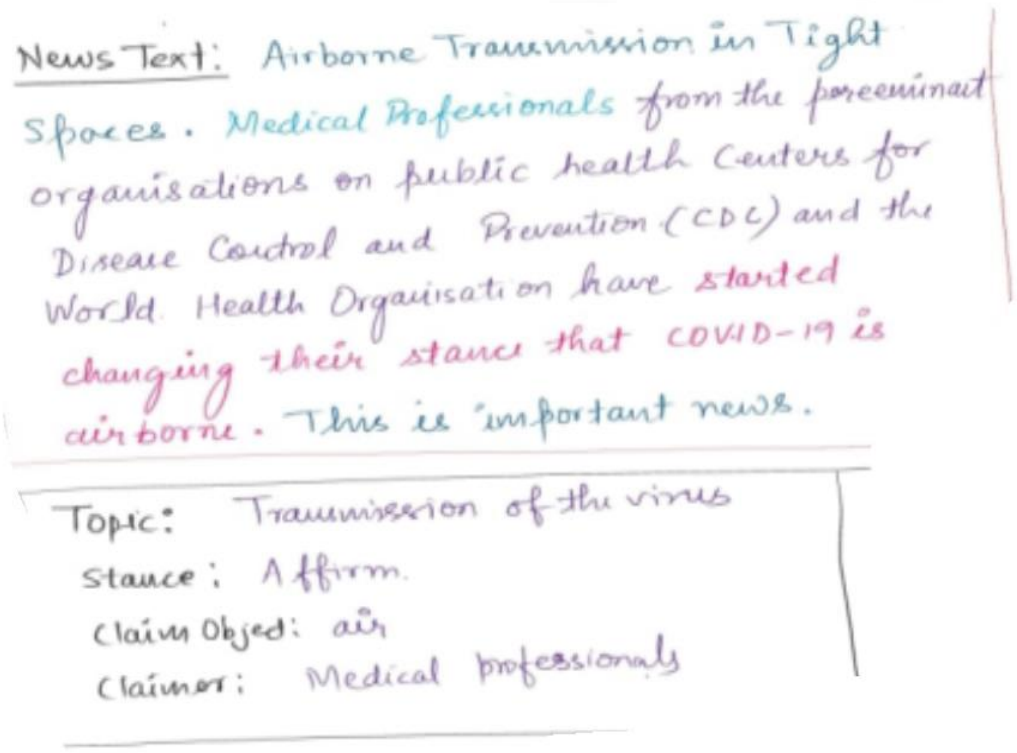
ANNOTATORS INTERFACE PHASE-1

Displays entire news article with target sentence highlighted in red (annotators are asked whether the highlighted sentence contains claim related to 4 predefined topics

ANNOTATORS PHASE-2

Identify the claim span, claim object and claimer and claimer stance from news article

In the developement set of the NEWSCLAIMS DATASET the data annotated as defined is used for few shot-learning training the model hyperparameters.



Chapter 4

Baselines and trainings defined in Research paper assigned

In this describing various zero-shot and prompt-based few shot learnings for detecting each attribute.
Sentences that just assert evidences or presents fact should not be considered as claims.
Factually verifiable claims(based on checking worthiness estimation).

Claim Sentence Detection:

Detecting all sentences in the claim that are related to pre-defined topics**.**

Step 1: ClaimBuster:

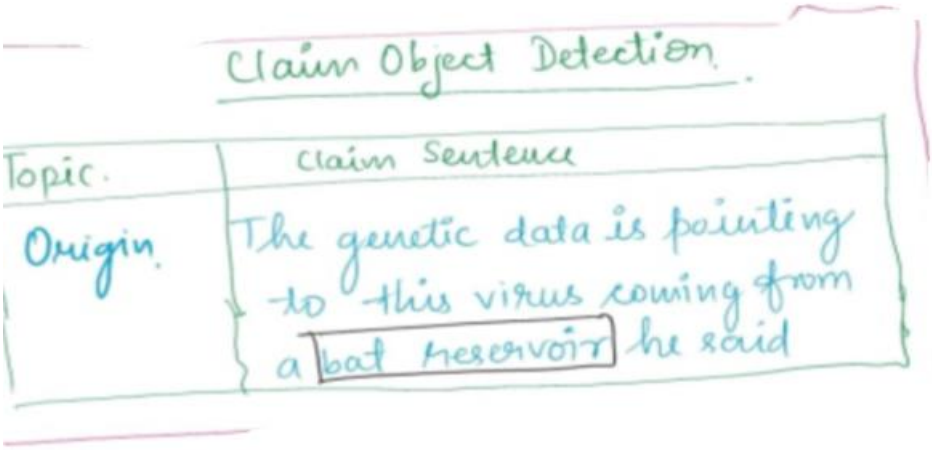
It is a claim-spotting system that is trained in dataset of check worthy claims.
It does not have knowledge about topics, so zero-shot is used.

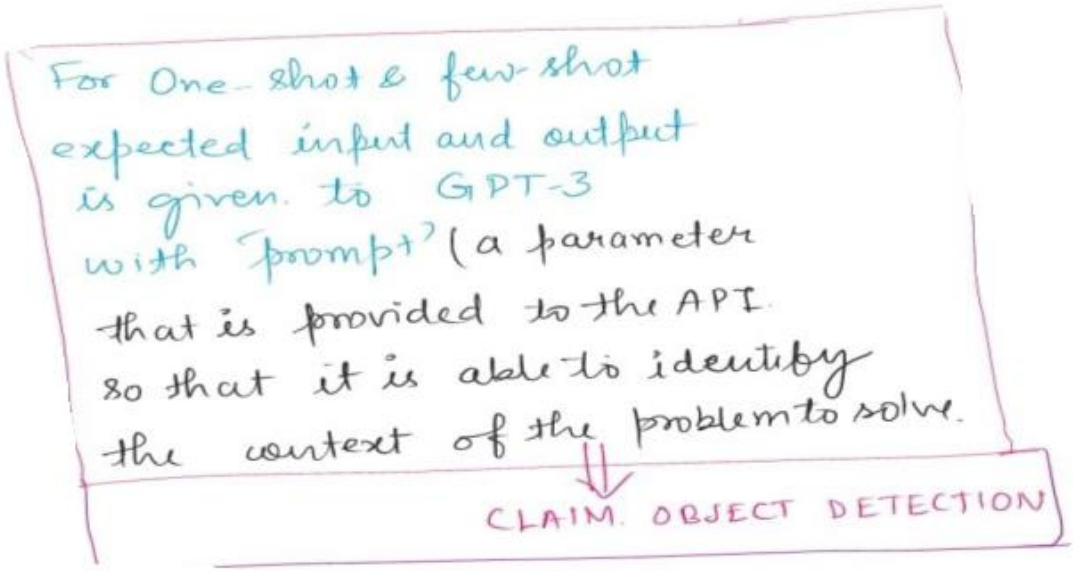
Step 2: ClaimBuster+Zero-shot NLI:

Pre-trained NLI models are used as zeo-shot text classifiers.
Claim sentences are classified as the NLI premise.
Entailment score of each topic is used to choose the relevant topic.

Claim Object Detection:

Claim sentence and topic are given, it identify what is being claimed about the topic in sentence.





Both zero-shot and few-shot learnings can be explored.

In-context learning(few-shot):

We have a pre-trained model and examples are inserted into it.

Prediction to be made is included as prompt at the end of contect.

GPT-3 is used as the language model in this.

Prompt-based fine-tuning (few-shot):

Pre-trained model is fine tuned to learn from labeled examples.

Examples are converted into a prompt.

It generates target text that will be replaced with <MASK> token**.**

Then, claim object is generated from <MASK> **token.**

Prompting(zero-shot):

Previous language model is used but in zero-shot settings.
GPT-3 is not provided with labeled examples.

Stance Detection:

It specifies that if claimer is asserting or refuting the claim.

Zero-shot NLI:

Hypothesis is constructed in this for affirm and refute labels.

Stance is taken based on entailment score.

Claim Span Detection:

It identifies exact claim boundaries within the sentence.

Debater Boundary Detection:

This service is used from Project Debater API.

This is based on BERT-large**.**

PoINeAR-Content:

It is a news attribution corpus of annotated triplets.

It contains *source, cue and contents* for statements made.

It is formed by fine-tuning BERT-large to identify content span with start and end classifier.

Claimer Detection:

This task identify if claim is made by journalist or reported source.

PoINeAR-Source:

News article is used as input.

Special token are used to mark content span..

Source annotation is used as the claimer and mark the content span within statement using tokens.

Source span is extracted using fine-tuned BERT-Large model.

Source span has start and end classifier.

For evaluation sum of start and end classifier is used as confidence score.

This threshold is to check if claim is by journalist.

SRL:

Semantic Role Labeling for claimer extraction.

It outputs verb predicate-argument structure of a sentence (who did what to whom).

Claim sentence is given as input.

Verb predicates are filtered that match pre-defined cues.

Verb predicates are filtered that match pre-defined cues.

SRL works at sentence level, so it cannot extract claimer outside the sentence.

So, if there is no verb predicate matching it outputs journalist

Chapter 5

Experiments as per Research Papers Guidelines

Model	P	R	F1
ClaimBuster	13.0	**86.5**	22.6
ClaimBuster + Zero-shot NLI	**21.8**	53.3	**30.9**
Human (single)	52.7	70.0	60.1
Human (3-way majority voting)	60.2	83.5	70.0

Table 2: Performance (in %) for various systems for detecting claims related to COVID-19.

Approach	Model	Type	F1
Prompting	GPT-3	Zero-shot	15.2
Prompting	T5	Zero-shot	11.4
In-context learning	GPT-3	Few-Shot	**51.9**
Prompt-based fine-tuning	T5	Few-Shot	51.6
Human	-	-	67.7

Table 3: F1 score (in %) for various zero-shot and few-shot systems for the claim object detection sub-task.

Model	Prec.	Recall	F1
PolNeAR-Content	67.0	42.8	52.3
Debater Boundary Detection	**75.7**	**77.7**	**76.7**
Human	82.7	90.9	86.6

Table 5: Performance (in %) of different systems for identifying the boundaries of the claim.

Model	Affirm F1	Refute F1	Acc.
Majority class	82.5	0.0	70.3
NLI (no topic)	89.1	68.0	83.8
NLI (with topic)	**91.1**	**78.8**	**87.5**
Human	97.0	84.2	94.9

Table 4: F1 score (in %) for *affirm* and *refute* classes along with the overall accuracy for stance detection. Zero-shot NLI is shown separately based on access the topic while constructing the hypothesis.

Chapter 6

Our Contribution Codes & Results & proposed novel ideas

- **At the provided github link No codes were there for language model pretraining**
- **Only codes related to preparation for data sets and calculation of recall precesion and F1 score were available**
- **We written the codes for pretraining for all other tasks .**

The claim sentence detection task which did not required any pretrained outputs gave the result as

For other remaining tasks (Claimer detection, claimer stance detection, claim span detection ,and claim object detection)

To make predictions on news claim text files using the fine-tuned model on the preprocessed data and save the predictions to an output file.
We used the following code

```
# Load the input data files
with open(ALL_SENTS_FILE) as f:
    all_sents = json.load(f)

with open(DEV_FILE) as f:
    dev_data = json.load(f)

with open(TEST_FILE) as f:
    test_data = json.load(f)

# Preprocess the input data
all_sents_inputs, all_sents_labels = preprocess(all_sents)
dev_inputs, dev_labels = preprocess(dev_data)
test_inputs, test_labels = preprocess(test_data)

# Fine-tune the model on the preprocessed data
model.train()
optimizer = torch.optim.AdamW(model.parameters(), lr=2e-5)

for epoch in range(3):
    optimizer.zero_grad()
    loss, _, _ = model(all_sents_inputs["input_ids"], all_sents_labels)
    loss.backward()
    optimizer.step()
```

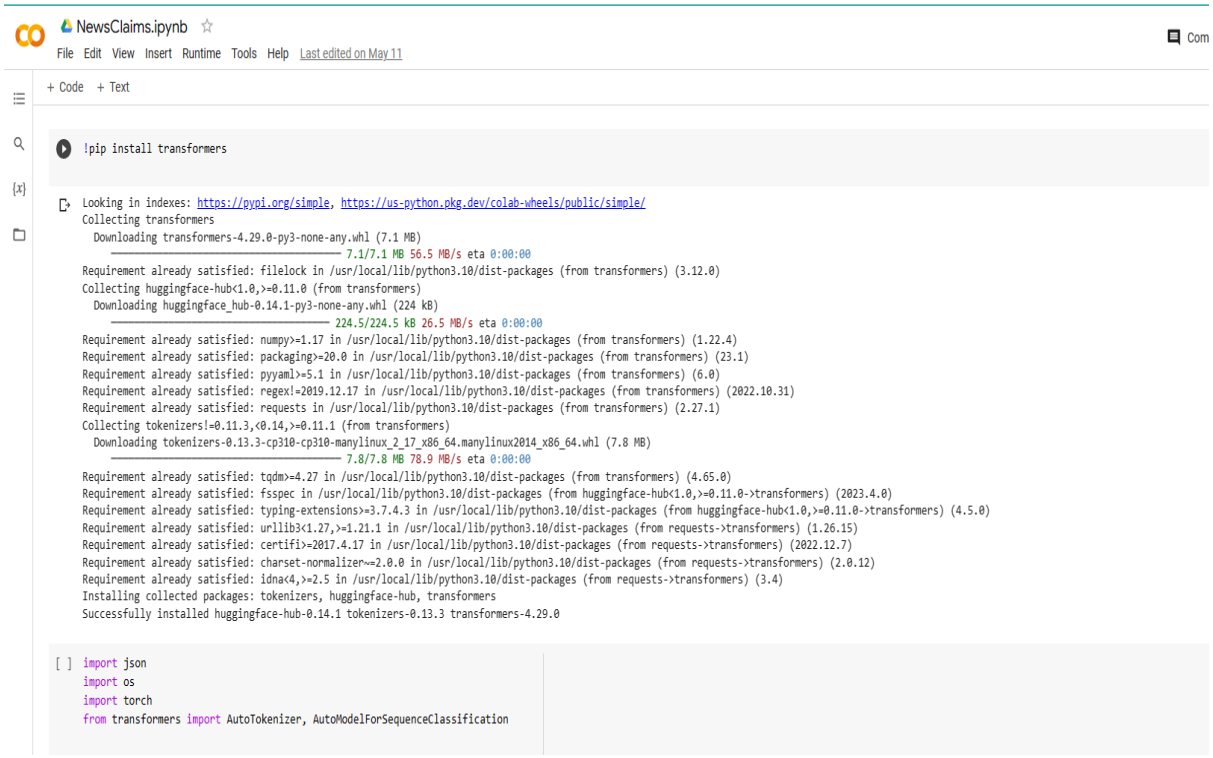
For output prediction we could not collect the result in prediction.json file as model could not be trained (due to limited machine capability) which has been used in the function provided by github repository to calculate precesion, recall and F1 score for various tasks.

Model was being trained on `ALL_SENTS_FILE`

Following error occurred due to insufficient machine capability
While making prediction on news claim text file

```
# Make predictions on the news claim txt files
model.eval()
predictions = {}

for file_name in os.listdir(NEWS_CLAIMS_DIR):
    with open(os.path.join(NEWS_CLAIMS_DIR, file_name)) as f:
        claim = f.read().strip()

    inputs = tokenizer(claim, padding=True, truncation=True, return_tensors="pt")
    preds = predict(model, tokenizer, inputs)

    predictions[file_name] = "Deny" if preds.item() == 0 else "Affirm"

# Write the predictions to the output file
with open(PREDICTION_FILE, "w") as f:
    json.dump(predictions, f)
```

```
Downloading (…)okenizer_config.json: 100% 48.0/48.0 [00:00<00:00, 2.09kB/s]
Downloading (…)lve/main/config.json: 100% 629/629 [00:00<00:00, 15.3kB/s]
Downloading (…)solve/main/vocab.txt: 100% 232k/232k [00:00<00:00, 6.54MB/s]
Downloading pytorch_model.bin: 35% 94.4M/268M [00:01<00:01, 99.0MB/s]
```



NOVEL IDEAS PROPOSED

- Instead of taking claimers stance(1) as affirm or refute(0) we can rate the claimers stance from 1-10 scale
- Another direction is build a unified framework that can extract claims and corresponding attributes together, without the need for separate components for each attribute.

CONCLUSION AND FUTUTRE SCOPE

We showed that zero-shot and prompt-based few-shot approaches can achieve promising performance in such low-resource scenarios, but still lag behind human performance, which presents opportunities for further research. In future work, we plan to explore extending this to build claim networks by identifying relations between the claims, including temporal connections. Another direction is build a unified framework that can extract claims and corresponding attributes together, without the need for separate components for each attribute