ISSN 2007-9737

# Towards Inclusive Fact-Checking: Claim Verification in English, Hindi, Bengali, and Code-Mixed Languages

Pritam Pal[1,*], Shyamal Krishna Jana[1], Arpan Majumdar[2], Dipankar Das[1]

[1] Jadavpur University, Kolkata, India

[2] University of Kalyani, Kalyani, Nadia, India

{pritampal522, shyamalkrjana516, arpanmajumdar952, dipankar.dipnil2005}@gmail.com

**Abstract.** Automated claim verification has gained significant attention in recent years due to the widespread dissemination of misinformation across various digital platforms. While substantial progress has been made for high-resource languages like English, claim verification for low-resource languages and specifically for Code-Mixed texts remains largely unexplored in a multilingual country like India. In the present work, we introduce a novel multilingual dataset for claim verification, covering English, Hindi, Bengali, and Hindi-English Code-Mixed languages. The dataset is developed by engaging large language models (LLMs) as well as human annotators. The dataset contains claims, evidence passages, and veracity labels (*SUPPORTS* or *REFUTES*) on news headlines collected from three important domains (Politics, Healthcare, Law and Order). We proposed a rule-based baseline algorithm and a dual-encoder framework based on transformer models to effectively verify claims across diverse languages. Our results show that XLM-RoBERTa achieves the best performance for English and Code-Mix texts, while IndicBERTv2 outperforms for Hindi and Bengali, respectively. This study highlights the challenges and opportunities in multilingual and Code-Mixed claim verification, offering a step towards building inclusive, language-diverse fact-checking systems even for low resource setup.

**Keywords.** Claim verification, fact checking, low-resourced language, prompt engineering.

## 1 Introduction

Spreading of misinformation and disinformation through online platforms has become a significant global issue. Automated claim verification systems, which aim to verify whether a given claim is true based on available evidence(s), are crucial tools in combating this issue. In recent years, considerable research has been conducted on claim verification, particularly for the English language. Many datasets, such as FEVER [25], SciFact [28], and COVID-Fact [19], have helped researchers to improve models using advanced techniques like transformer-based architectures (e.g., BERT [6], RoBERTa [4]).

While most of such attempts focus on resource heavy languages, such as English, there is a significant research gap observed for low-resource languages, particularly Indian languages like Hindi and Bengali. Millions of people speak these languages worldwide, including the majority of people in India and Bangladesh [1]; however, there are very few annotated datasets and models available to support claim verification or related tasks. As a result, misinformation in these communities often remains unchecked.

On the other hand, even more challenging area is to verify claims in Code-Mixed languages, where people use a mix of two languages (for example, Hindi and English) in the same sentence. This

is very common in social media and in informal conversations, but it introduces many linguistic challenges, such as inconsistent grammar, mixed vocabulary, and transliteration issues. Despite the growing importance of Code-Mixed communication, claim verification for Code-Mixed texts remains largely unexplored.

To address the above mentioned challenges, the present research contributes on building a multilingual and Code-Mixed dataset for claim verification. We target four language settings for our study: English, Hindi, Bengali, and Hindi-English Code-Mix. Employing a combination of large language models (LLMs) and human annotation, we prepared a dataset containing claims, evidence, and their veracity labels: *SUPPORTS* and *REFUTES*. Along with the dataset, two claim verification frameworks have been developed: one is based on simple rule-based approach by utilizing the Smith-Waterman sequence alignment algorithm [23] while another one is a dual-encoder framework based on pre-trained transformer models. The overall contributions in this paper can be summarized as follows:

— A novel dataset has been prepared for claim verification with around 2.7K claim-evidence pairs for English, Hindi, Bengali, and Hindi-English Code-Mix languages using various LLM prompts followed by a manual human annotation process.

— A comparatively lightweight claim verification framework is developed using the Smith-Waterman sequence alignment algorithm.

— Further, a dual-encoder multilingual claim verification framework was developed utilizing state-of-the-art transformer models.

— Finally, a comprehensive analysis of the results obtained from these models was investigated, providing valuable insights into their performances.

The remainder of the paper is organized as follows: Section 2 reviews prior research related to claim verification and fact-checking. Section 3 provides a detailed description of the dataset, including a fully automated approach that utilizes LLM and a semi-automated approach with LLMs and manual annotation. Section 4 outlines the overall methodology for the claim verification work, including the training process on the developed dataset and the hyperparameter settings. This section also details the development of the pre-trained transformer-based multilingual dual-encoder framework, as well as a rule-based claim verification approach that employs the Smith-Waterman sequence alignment algorithm. The experimental results for both the Smith-Waterman algorithm and the dual-encoder multilingual transformer-based approach are discussed in Section 5. Finally, Section 6 concludes the draft by providing a summary of the key findings, and Section 7 addresses potential limitations and outlines future research directions for the present study.

## 2 Related Work

Claim verification, a crucial task in combating misinformation, has gained significant attention in recent years. Several researchers have proposed various methodologies and datasets related to claim verification. [25] introduced the FEVER dataset, laying the groundwork for automated fact extraction and verification tasks. [10] proposed another claim verification dataset with natural claims in different domains with a sample size of around 6.4K. The authors demonstrated the effectiveness of the BERT [6] model in claim verification tasks, emphasizing the importance of evidence selection and model fine-tuning. [11] proposed the 'HOVER' dataset, which is a multi-hop fact extraction and claim verification dataset with 2, 3, and 4 hops. [3] proposed a technique where they verify claims using a question-answer pair as evidence. [16] proposed a zero-shot-based fact verification framework by generating questions from evidence. [22] proposed the 'AVERITEC' dataset for real-world claim verification by collecting evidence from the web. [30] presented a claim verification method that can verify complex claims and generate their explanations without the help of any evidence

using Large Language Models (LLMs). The authors reported their approach outperforms the previous works on 'HOVER', 'FEVEROUS' [2] and 'SciFact-Open' [28] datasets. [8] also proposed a LLM-based claim verification method using GPT 3.5 and GPT 4 models.

Regarding claim verification in scientific documents, several studies have been conducted by various researchers. [26] proposed the 'SCIFACT' dataset with 1.4K scientific claims and their evidence to verify scientific claims. [18], [33] and [29] also utilized the 'SCIFACT' dataset in their study to verify scientific claims. Along with 'SCIFACT' dataset, [29] used other datasets such as 'HelthVer' [21], 'COVIDFact' [19] etc. for scientific claim verification. [27] organized a shared task 'SCIVER' for scientific claim verification, where a total of 11 teams participated, and the best-performing team achieved an F1 score of 64.4 in sentence-level claim verification. [28] presented 'SciFact-Open', a comparatively large dataset with 500K samples for scientific claim verification.

In the medical domain, [24] proposed BM25 and BART-based models to verify COVID-19-related claims. [31] used entity and relation properties to verify biomedical claims. [17] performed their fact-checking task using Google Fact Check Tools in COVID-19 domains.

According to existing literature, claim verification has been extensively studied in scientific and medical domains, primarily in high-resource languages such as English. There's limited research on low-resource languages, particularly Indian languages, and especially in code-mixed contexts. This study addresses this gap by investigating claim verification in Hindi, Bengali, code-mixed, and English.

## 3 Dataset Preparation

We developed an entirely new dataset for the claim verification work using state-of-the-art LLMs and advanced prompt engineering techniques.

### 3.1 LLM-based Approach

The only LLM-based approach entirely relies on LLMs for dataset development. Specifically, the LLM was prompted to generate claims, identify corresponding evidence, and annotate each instance with a veracity label: *SUPPORTS* (if the evidence contains proper information that supports the claim as accurate) or *REFUTES* (if the evidence contradicts the information in the claim). We employed a few-shot prompting technique using the GPT-3.5-Turbo [14] model to generate the claims, associated evidence, and corresponding veracity labels. The following prompt was utilized during the dataset development process:

**Task:** Dataset Creation for Claim Verification.

**Goal:** Generate a dataset for claim verification tasks across various topics, ensuring unique claims with corresponding evidence and labels

****Fields****

**Claim:** The statement to be verified.

**Evidence:** Supporting evidence (text, links, documents) for the claim.

**Source of Evidence:** Source for the evidence of the claim. The sources are reputed public/private organizations' URLs, universities, journals, news articles etc.

**Label:** either SUPPORTS or REFUTES

****Instructions****

**Topic:** [Topic]

**Generate Claims:** For each topic, craft unique claims that can be either true or false. Ensure claims cover a spectrum of complexity and relevance.

**Gather Evidence:** Find or create supporting evidence (articles, research papers, official statements, etc.) for each claim. The evidence should clearly support or refute the claim.

**Assign Labels:** Based on the provided evidence, assign labels to each claim.

**Ensure Uniqueness:** Verify that all claims within the dataset are distinct and do not overlap in content or meaning.

**Format:** Organize the dataset in a structured JSON format with the specified fields (Claim, Evidence, Label).

****Examples****

*[3 examples for each topic]*

Utilizing the prompt mentioned above, over 950 unique claims in the English language, their corresponding pieces of evidence, sources for the evidence, and their veracity labels were

created across five different topics: Healthcare, Technology, Indian Politics, Finance, and Indian Parliamentary Affairs. A few examples of such a claim evidence pair are provided in Figure 2, and the distribution of veracity labels for this data preparation approach is provided in Figure 1.

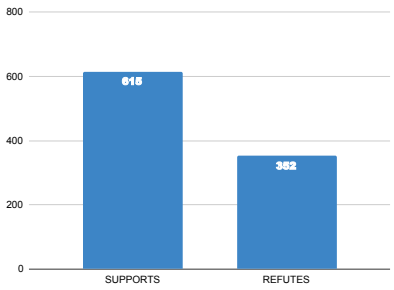


**Fig. 1.** Distribution of veracity labels in the LLM-only generated dataset

### 3.2 Semi-Automatic Approach

Since LLMs are prone to hallucinating, relying solely on the LLM-based data preparation approach may not yield robust claim verification frameworks, particularly in low-resource languages. Therefore, a partially manual or semi-automated approach was employed.

#### 3.2.1 Collection of Claim

The first phase of the partially manual annotation approach involved collecting claim texts. News headlines were identified as one of the most effective sources for extracting claim sentences.

To gather these headlines, we employed a web scraping technique using Python's BeautifulSoup library. Headlines were scraped from various online news portals, including *Aaj Tak*, *Dainik Jagran*, *India TV*, *Live Law*, *Aaj Tak Bangla*, and *Sangbad Pratidin*. During the scraping process, we focused on headlines related to the topics of *Politics*, *Healthcare*, and *Law and Order*.

Table 1 provides which news portal was used for which language in the news headline collection.

Since there are no news portals on the internet that provide Hindi-English Code-Mix texts, we utilized a subset of the dataset introduced by Nayak et al. [13], which comprises real-world Hindi-English Code-Mix tweets in Romanized script.

For Hindi, English, and Bengali news headline crawling, our focus was on topics related to *Politics*,

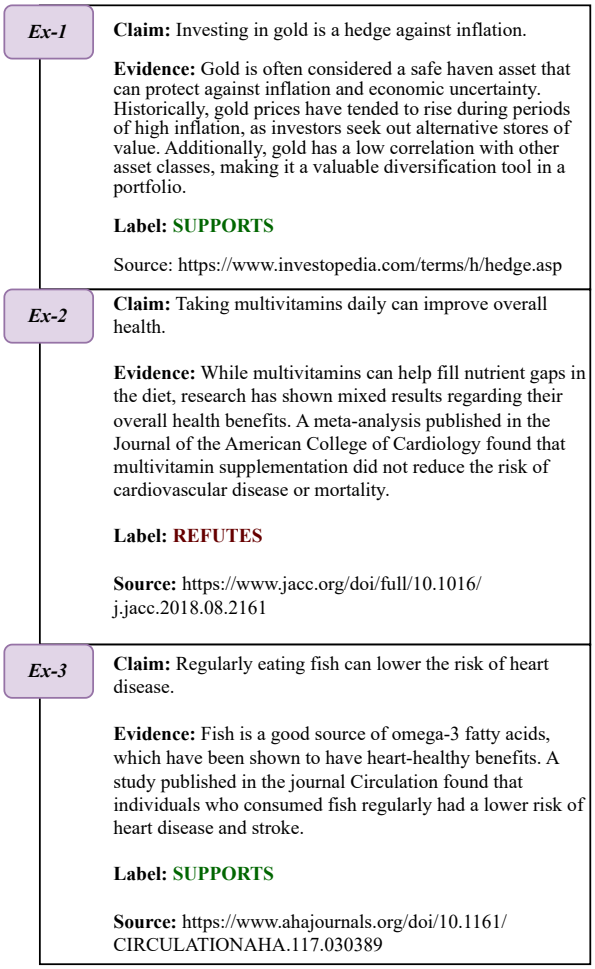


**Fig. 2.** Example of claim, evidence, veracity label, and the source of evidence for LLM-generated data

**Table 1.** Names of the online news portals for different languages used for data collection

| Language | News Portal |
|---|---|
| English | Live Law, Aaj Tak, India TV |
| Hindi | Dainik Jagaran, Live Law |
| Bengali | Aaj Tak Bangla, Sangbad Pratidin |

*Healthcare*, and *Law and Order*.

To ensure consistency, we also filtered the Code-Mix data to retain only those tweets relevant to these domains, employing the Llama 3.2 72B [9] LLM model for topic filtering. The following prompt was provided to LLM model to filter out the topics:

You are an efficient language model that can precisely find a topic in a Hindi-English code-mixed text (mixing of Hindi and English in romanized script). Now, provide the topic of 'text' in the English language. The topics should be in between 'politics', 'indian election', 'national', 'hindu-muslim', 'sports', 'entertainment', 'unknown'. Remember, only generate the exact output. No extra explanation is needed.

### 3.2.2 Collection of Evidence

After gathering claims in English, Hindi, Bengali, and Hindi-English Code-Mix, we retrieved supporting evidence for each claim using state-of-the-art LLMs. Acknowledging the potential biases that LLMs might introduce during the evidence collection process, we utilized three advanced LLMs: GPT-4o-mini [15], DeepSeek-v3 [5], and QWEN2-72b-Instruct [32], rather than depending on a single model. This multi-model strategy was implemented to enhance the reliability and objectivity of the collected evidence. Each LLM was given the following prompt for the evidence collection:

You are an efficient language model that generates precise evidence for a given news headline: "{claim}". Provide a single, well-structured paragraph in {lang} without repetition. Ensure the response is within 120 words and does not contain extra newlines or unnecessary formatting. Do not include quotation marks around the response.
Now, generate the evidence in {lang} with these constraints.

### 3.2.3 Manual Annotation

We employed five postgraduate students from the Linguistics department, who had proficiency in reading and writing English, Hindi, and Bengali, as well as Code-Mix texts, for the further annotation process. The annotators were instructed to — 1) Find a piece of evidence for the given claim from reputed sources such as reputed news articles, scientific reports or journals, government documents, etc. along with its source, 2) Identify best evidence from the given four pieces of evidences (one is manually collected and three were LLM generated), and 3) annotate the claim-evidence pair with *SUPPORTS* or *REFUTES* labels. To remove the bias from the annotator's mind when choosing the best evidence based on LLM models, we anonymized the evidence generated by each LLM model. The following instructions were given to annotators to identify the best evidence:

— **Context awareness:** The evidence should be contextually relevant to the claim. Verify that the response accurately represents the context and does not oversimplify complex topics.

— **Detailing and Depth in the evidence:** The evidence should be quite detailed, explainable and contain proper information. Brief evidence should be rejected. Look for well-structured arguments rather than generic responses.

— **Source of Evidence:** Pick the evidence where proper sources or references are mentioned. Verify whether the sources are real or not.

— **Objectivity and Bias:** The evidence should be neutral and should not contain any political or other type of bias.

— **Logical Consistency:** Select the evidence that logically supports/ refutes the arguments. Avoid pieces of evidence that jump between unrelated ideas and lack structured reasoning.

— **Verify Named Entities, Dates, and Numbers in the evidence:** An evidence contains factual information such as named entities, dates, or numbers (if any). The evidence can be considered good evidence. However, the annotators cross-check the named entities, dates, or numbers in the evidence if available.

After selecting the best evidence, the annotators were asked to annotate the claim-evidence pair as *SUPPORTS* or *REFUTES* labels based on the information in the best annotated evidence.

***Reason for manual evidence collection by annotators:*** During the best evidence and veracity annotation process, annotators were instructed to identify evidence for each claim exclusively from reputable sources manually. This protocol was implemented to address the

propensity of LLMs to hallucinate during evidence extraction, which can result in the generation of inaccurate or inappropriate evidence, specifically in low-resourced languages such as Bengali. To ensure the reliability and credibility of the collected evidence, annotators were explicitly directed to source information from authoritative outlets, including government websites, scientific journals, and official reports. This approach was designed to enhance the quality and factual accuracy of the evidence, mitigating the limitations associated with automated extraction methods in these linguistic contexts.

#### 3.2.4 Refute Evidence Generation

Since the news headlines in English, Hindi, and Bengali were collected from reputable news sources, the majority of claims in the news headlines are supported by their best annotated evidence. Therefore, after manual annotation, there was a high level of class imbalance between *SUPPORTS* and *REFUTES* labels. Using this imbalanced dataset to train our claim verification framework results in a biased framework for the majority class label. To overcome this, we modified the pieces of evidence that support its claim in such a way that the claim refutes the modified evidence. We employed the GPT-4o-mini with prompting to perform this task. The following prompt was utilised during the modification of proof:

You have given a claim and its corresponding evidence. Note that, the claim supports its evidence.

Now modify the evidence in such a way that the claim refutes its evidence.

**CLAIM:** {claim}

**EVIDENCE:** {evidence}

Remember, only generate the evidence. No extra explanation is needed.

Finally, we have a total of 597, 410, 489, and 321 annotated claim-evidence pairs for English, Hindi, Bengali, and Code-Mix languages, respectively. Two examples of supported claims, their supported evidence, and LLM-generated refuted evidence are provided in Figure 3.

Next, 50% of the data from these datasets was divided and used for testing purposes. The remaining 50% of the data was combined and used for training purposes. However, the dataset that was entirely generated by the GPT-3.5-Turbo model in the 'only LLM-based approach' was not included in the test split due to its artificial nature. This exclusion ensures a robust evaluation against real-world data patterns, as synthetic claims may exhibit systematic biases or artifacts that are unrepresentative of authentic claims. The GPT-3.5-Turbo-generated data was instead fully utilized for training purposes to leverage its scalability while maintaining test-set integrity. The overall language-specific train-test split is provided in Table 2.

**Table 2.** Distribution of data for training and testing

| | | Labels | |
|---|---|---|---|
| | **Langauge** | **SUPPORTS** | **REFUTES** |
| Train | English | 758 | 508 |
| | Hindi | 95 | 113 |
| | Bengali | 117 | 128 |
| | Code-Mix | 73 | 86 |
| Test | English | 155 | 142 |
| | Hindi | 109 | 93 |
| | Bengali | 127 | 117 |
| | Code-Mix | 85 | 77 |

## 4 Methodology

This section provides a brief description of the methodology used to identify whether a claim supports or refutes its evidence. Given a claim $C$ and evidence $E$, our main objective is to develop a framework that can identify its veracity label as 'SUPPORTS' or 'REFUTES'. To develop the framework, two approaches were employed: one utilized the rule-based Smith-Waterman sequence alignment algorithm, and the other used a dual-encoder framework with pretrained multilingual transformer models.

### 4.1 Rule-based Sequence Alignment Algorithm

In this section, we present a sequence alignment-based algorithm designed for fact-checking and veracity identification. This algorithm offers a cost-effective alternative to deep learning models for identifying the most

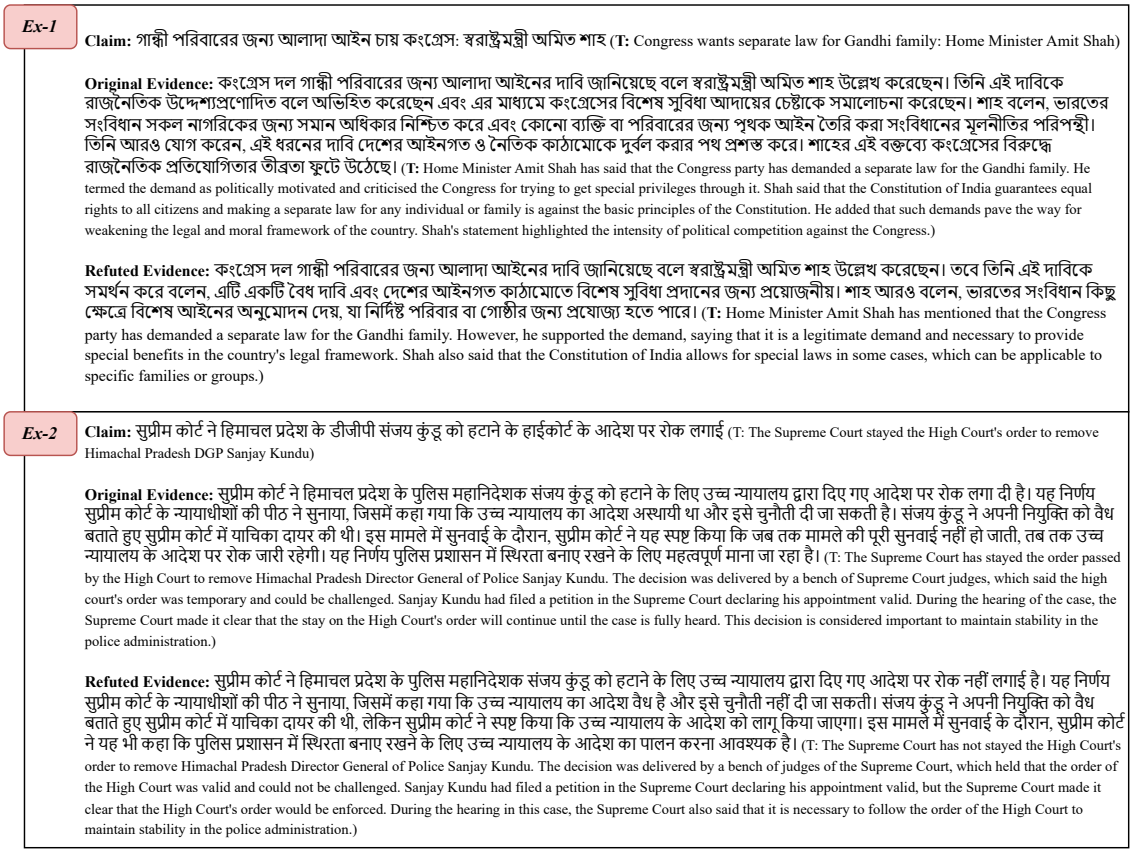


**Fig. 3.** Example of claim, their supporting evidence (identified by annotators), and LLM-generated refuted evidence.

reliable evidence to support a claim, significantly reducing both instrumental and experimental expenses. We implement this approach using the "Smith-Waterman" sequence alignment algorithm, a well-established local sequence alignment method.

The Smith-Waterman algorithm is a dynamic programming technique used for determining local sequence alignments. It assesses the similarity between two biological sequences, such as DNA and RNA. By comparing two sequences, the algorithm identifies the optimal local alignment score.

Before moving on to the algorithmic part, we first perform some preprocessing on the input text, which includes both the claim and evidence pairs. The initial step involves cleaning the text by removing unwanted symbols, special characters, and punctuation to reduce noise and simplify the input. The second and most crucial step is tokenization of the input text. We use an index tokenization approach to convert words in the input text into their corresponding numerical indices. This process consists of several steps: First, we combine all the words from the claim and evidence into a single sentence. Next, we assign a unique index to each of the combined words. Finally, we extract the indexed tokens for the claim and evidence separately based on their positions in the original claim and evidence pair.

The Smith-Waterman algorithm has three primary parameters: the match score, the mismatch penalty, and the gap penalty. For the languages used in our study, the match score was set to specific values: 2.5 for English, 3 for Hindi, 3 for Bengali, and 8 for Code-Mix. These match

scores were established through trial and error using our training data. This score is represented by the variable $lagnguage_{match}$. In cases of mismatches or gaps between sequence pairs, the algorithm deducts a small fixed amount from the overall score, with both gap and mismatch penalties uniformly set to -1 across all language types. The final score is then stored in a designated variable.

To begin, we execute Algorithm 1 to obtain the $maxscore$ from the Smith-Waterman algorithm, as evaluated in Algorithm 2. We then apply a threshold value ($\tau = 0.5$) to classify the predicted score ($score_e$) and derive the predicted label for the claim.

Finally, we assess the algorithm's performance by comparing these predicted labels with the original labels.

**Algorithm 1** SmithWaterman$_{\text{Modified}}$

**Require:** $seq_1$, $seq_2$
$m \leftarrow size(seq_1)$, $n \leftarrow size(seq_2)$
$scoring_{mat}[m+1][n+1]$ *of floating value*
$match \leftarrow lagnguage_{match}$, $gap \leftarrow -1$, $mismatch \leftarrow -1$, $score \leftarrow 0$, $maxscore \leftarrow 0$
**for** *i from 0 to (m+1)* **do**
  $scoring_{mat}[i][0] \leftarrow 0$
**end for**
**for** *j from 0 to (n+1)* **do**
  $scoring_{mat}[0][j] \leftarrow 0$
**end for**
**for** *i from 0 to (m+1)* **do**
  **for** *j from 0 to (n+1)* **do**
    **if** $seq_1[i-1] = seq_2[j-1]$ **then**
      $score \leftarrow match$
    **else**
      $score \leftarrow mismatch$
    **end if**
    $scoring_{mat}[i][j] \leftarrow max(0, scoring_{mat}[i][j-1] + gap, scoring_{mat}[i-1][j] + gap, scoring_{mat}[i-1][j-1] + score)$
    **if** $maxscore \leq scoring_{mat}[i-1][j-1]$ **then**
      $maxscore \leftarrow scoring_{mat}[i][j]$
    **end if**
  **end for**
**end for**
**return** *maxscore*

Time complexity of the Smith-Waterman algorithm is O(m*n) and space complexity is O(m*n), where m stands for the length of the claim and n stands for the length of the evidence.

**Algorithm 2** Claim Verification with Evidence scoring

**Require:** dataset
**Require:** $score_e[size(dataset)]$ *of floating values*
$org \leftarrow dataset[label]$
**for** $i$ *from* $0$ *to* $size(dataset)$ **do**
  $c_t, e_t \leftarrow Tokenize(dataset[claim][i], dataset[evidence][i])$ ▷ [word index tokenization]
  $score \leftarrow SmithWaterman_{Modified}(c_t, e_t)$
  $score_e[i] \leftarrow \frac{score}{size(c_t)}$
  $i \leftarrow i + 1$
**end for**
**for** $i$ *from* $0$ *to* $size(score_e)$ **do**
  **if** $\tau \leq score_e[i]$ **then**
    $pred \leftarrow 1$
  **else**
    $pred \leftarrow 0$
  **end if**
**end for**
$prec \leftarrow Precision\_Score(pred, org)$,
$rec \leftarrow Recall\_Score(pred, org)$
$f1 \leftarrow F1\_Score(pred, org)$

### 4.2 Dual Encoder Framework with Pre-trained Transformer

The dual-encoder framework utilised the state-of-the-art multilingual transformer models, such as lightweight DistilBERT-multilingual [20] to multilingual-BERT [6], XLM-RoBERTa [4] to Indian language-focused models like MuRIL [12] and IndiBERTv2 [7]. The overall system framework for this approach is depicted in Figure 4.

#### 4.2.1 Tokenization

Before going to framework development and training of the frameworks, the input text (claim and evidence) was tokenized into a sequence of tokens. Since the claims are comparatively shorter sentences than the evidence, which is generally longer text, the input claims were tokenized to a fixed sequence length of 128 tokens, and the input evidences were tokenized to a fixed sequence length of 512 tokens. The tokenization was performed using the tokenization method of the corresponding transformer model. For

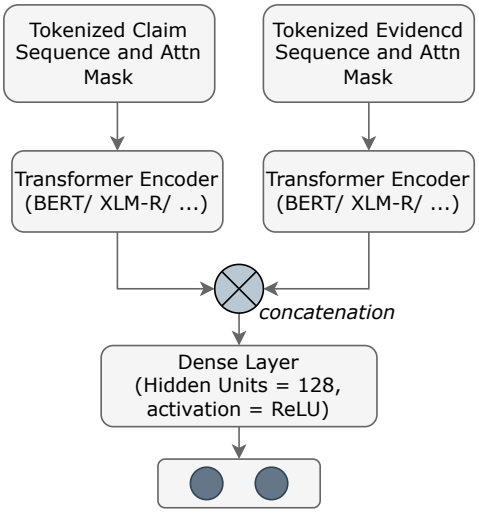


**Fig. 4.** Flow diagram of the dual-encoder framework (one encoder for claim and another for evidence) using pre-trained multilingual transformer models: mDistilBERT, mBERT, XLM-RoBERTa, MuRIL, and IndicBERTv2

example, the DistilBERT tokenizer was used for DistilBERT-multilingual, and the BERT tokenizer was used for multilingual-BERT, and so on. All the tokenizers return a sequence of tokens and attention masks, which were further provided as input to the transformer encoders.

#### 4.2.2 Model Selection

As previously mentioned, to develop the frameworks, several pre-trained transformer models were utilized, including DistilBERT-multilingual, multilingual-BERT, XLM-RoBERTa, IndicBERTv2, and MuRIL. All these models are trained on a diverse set of languages and are capable of processing and understanding our input languages, such as English, Hindi, Bengali, and Hindi-English Code-Mix. Moreover, MuRIL and IndicBERTv2 were trained explicitly on Indian languages, which enables them to understand Indian languages (e.g., Hindi, Bengali, and Code-Mix) more effectively than other models.

#### 4.2.3 Framework Description

The dual-encoder framework was developed to identify the veracity of a claim, where one encoder was used for the claim and another encoder was used for the evidence. In the first transformer encoder, the tokenized claim sequence and attention mask were provided as input, and in the second transformer encoder, the tokenized sequence of evidence and attention mask were provided as input, as shown in Figure 4.

Further, the pooled output from both transformer encoders, which is obtained by applying a learned linear transformation and $\tanh$ activation to the first token's ('[CLS]' for DistilBERT, mBERT, MuRIL and IndicBERTv2, and '<s>' for XLM-RoBERTa] model)[1] hidden representations were concatenated:

$$H^C = [h_0^C + h_1^C + h_2^C + ... + h_{127}^C] \in \mathbb{R}^{128 \times d},$$
$$H^E = [h_0^E + h_1^E + h_2^E + ... + h_{511}^E] \in \mathbb{R}^{512 \times d},$$
$$\mathbf{Pooler}^C = tanh(\mathbf{W}^C h_0^C + b^C),$$
$$\mathbf{Pooler}^E = tanh(\mathbf{W}^E h_0^E + b^E),$$
$$\mathbf{Z_{concatenate}} = \mathbf{Pooler}^C \otimes \mathbf{Pooler}^E,$$

where $H^C$ and $H^E$ represent the last hidden state output from the claim transformer encoder and the evidence transformer encoder, respectively. The $\mathbf{Pooler}^C \in \mathbb{R}^d$ and $\mathbf{Pooler}^E \in \mathbb{R}^d$ represent the pooled output from claim and evidence encoders, and $\mathbf{W}^C$ and $\mathbf{W}^E$ represent the trainable weight matrices, and $b^C$ and $b^E$ represent the bias vectors. The $\mathbf{Z_{concatenate}} \in \mathbb{R}^{2d}$ represents the concatenated output from the claim and evidence pooled output, and $d$ is the hidden size.

Next, the $\mathbf{Z_{concatenate}}$ was passed through a dense layer of 128 hidden units with the ReLU activation function.

$$\mathbf{Z_{dense}} = ReLU(\mathbf{Z_{concatenate}}),$$

where $\mathbf{Z_{dense}} \in \mathbb{R}^{128}$ represent the output of dense layer.

[1] '[CLS]' and '<s>' are the special tokens to their corresponding models which represent the starting token of a sequence.

### 4.2.4 Classification

For classification, the output of the dense layer ($\mathbf{Z_{dense}}$) was passed through the output layer of 2 hidden units. The output layer used softmax as its activation function.

$$\mathcal{P} = softmax(\mathbf{Z_{dense}}),$$
$$\hat{\mathcal{Y}} = \arg\max_{j}(\mathcal{P}),$$

where $\mathcal{P}$ represents the probability values for each class, $\hat{\mathcal{Y}}$ represents the class containing the maximum probability value, and $j$ is the number of classes.

### 4.2.5 Training

To accomplish the training process, the training data mentioned in the Table 2 was used. During the training of models, 90% of the training data was used for actual training, and the remaining 10% of the data was used as a validation set. The `SparseCategoricalCrossEntropy` loss function was used with a learning rate of 2e-5 (XLM-RoBERTa, MuRIL, and IndicBERTv2) and 3e-5(mDistilBERT and mBERT). The optimizer was chosen as AdamW, and all the frameworks were trained up to five epochs. The batch size was taken as 4 for XLM-RoBERTa, MuRIL, and IndicBERTv2 model-based frameworks, and 8 for mDistilBERT and mBERT model-based frameworks. The training loss vs validation loss curves for each transformer model are provided in Figure 5.

## 5 Experiment and Result

All the proposed frameworks were trained using TensorFlow modules on an NVIDIA A4000 GPU. The pre-trained transformer models and their corresponding tokenizers were utilized from the HuggingFace library[2]. To evaluate the performance of the frameworks, the precision, recall, and macro

[2]https://huggingface.co/

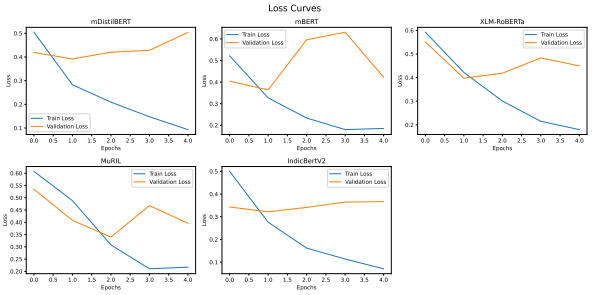


**Fig. 5.** Training loss vs Validation loss for each transformer model during training (From left to right: mDistilBERT, mBERT, XLM-RoBERTa, MuRIL, and IndicBERTv2)

F1-score were calculated for each language on the test data set.

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives + False Positives}},$$
$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives + False Negatives}},$$
$$\text{F1-Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision + Recall}}.$$

The overall performance and label-specific performances of the proposed framework for the Smith-Waterman algorithm approach and various transformer models are presented in Table 3.

From Table 3, for the English language, the XLM-RoBERTa model-based claim verification framework achieves the best performance among all evaluated frameworks, with precision, recall, and F1-scores of 86.92, 86.98, and 86.87, respectively. Furthermore, for label-specific results, the XLM-RoBERTa framework attains the highest F1-scores for both the *SUPPORTS* and *REFUTES* categories, with scores of 87.04 and 86.69, respectively.

The IndicBERTv2 model demonstrates superior performance for both Hindi and Bengali languages, achieving F1-scores of 80.20 and 85.65, respectively. Additionally, for label-specific evaluations, although IndicBERTv2 yields the highest F1-scores for both *SUPPORTS* and *REFUTES* labels, the DistilBERT-multilingual model achieves the best recall score of 89.91 for the *SUPPORTS* label. In the case of the *REFUTES* label for Hindi, the

**Table 3.** Overall performance and label-specific outcomes for claim verification system in rule-based Smith-Watrerman algorithm and different pretrained multilingual transformer models (Prec., Rec., and F1 are the abbreviations of Precision, Recall and F1-score)

| | | Overall Result | | | Label Specific Result | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | SUPPORTS | | | REFUTES | | |
| **Model** | **Language** | **Prec.** | **Rec.** | **F1** | **Prec.** | **Rec.** | **F1** | **Prec.** | **Rec.** | **F1** |
| Algorithm base approach | English | 68.41 | 68.41 | 68.41 | 68.46 | 68.69 | 68.57 | 68.37 | 68.14 | 68.25 |
| | Hindi | 69.81 | 69.47 | 69.38 | 67.72 | 75.89 | 71.58 | 71.90 | 63.04 | 67.18 |
| | Bengali | 71.08 | 70.62 | 70.55 | 68.90 | 77.66 | 73.02 | 73.25 | 63.57 | 68.07 |
| | Code-Mix | 56.34 | 56.40 | 56.27 | 61.11 | 56.54 | 58.74 | 51.56 | 56.25 | 53.80 |
| mDistilBERT | English | 85.26 | 85.30 | 85.18 | 88.28 | 82.58 | 85.33 | 82.24 | 88.03 | 85.03 |
| | Hindi | 71.24 | 65.92 | 64.82 | 64.47 | 89.91 | 75.10 | 78.00 | 41.94 | 54.55 |
| | Bengali | 77.62 | 76.21 | 76.20 | 73.33 | 86.61 | 79.42 | 81.91 | 65.81 | 72.99 |
| | Code-Mix | **78.74** | **78.07** | **78.15** | 76.60 | 84.71 | 80.45 | 80.88 | 71.43 | 75.86 |
| mBERT | English | 83.16 | 83.22 | 83.16 | 85.23 | 81.94 | 83.55 | 81.08 | 84.51 | 82.76 |
| | Hindi | 76.73 | 75.74 | 74.65 | 86.25 | 63.30 | 73.02 | 67.21 | 88.17 | 76.28 |
| | Bengali | 85.89 | 85.48 | 85.57 | 83.82 | 89.76 | 86.69 | 87.96 | 81.20 | 84.44 |
| | Code-Mix | 60.40 | 60.24 | 59.80 | 64.29 | 52.94 | 58.06 | 56.52 | 67.53 | 61.54 |
| XLM-R | English | **86.92** | **86.98** | **86.87** | 89.73 | 84.52 | 87.04 | 84.11 | 89.44 | 86.69 |
| | Hindi | 76.39 | 74.52 | 73.00 | 87.67 | 58.72 | 70.33 | 65.12 | 90.32 | 75.68 |
| | Bengali | 84.07 | 83.85 | 83.60 | 89.19 | 77.95 | 83.19 | 78.95 | 89.74 | 84.00 |
| | Code-Mix | 77.05 | 76.85 | 76.53 | 82.19 | 70.59 | 75.95 | 71.91 | 83.12 | 77.11 |
| MuRIL | English | 79.12 | 77.94 | 77.29 | 87.29 | 66.45 | 75.46 | 70.95 | 89.44 | 79.13 |
| | Hindi | 73.80 | 73.95 | 73.73 | 78.00 | 71.56 | 74.64 | 69.61 | 76.34 | 72.82 |
| | Bengali | 81.95 | 82.00 | 81.96 | 83.74 | 81.10 | 82.40 | 80.17 | 82.91 | 81.51 |
| | Code-Mix | 40.26 | 44.9 | 36.69 | 36.00 | 10.59 | 16.36 | 44.53 | 79.22 | 57.01 |
| IndicBERTv2 | English | 81.30 | 80.82 | 80.43 | 87.60 | 72.90 | 79.58 | 75.00 | 88.73 | 81.29 |
| | Hindi | **80.89** | **80.78** | **80.20** | 87.91 | 73.39 | 80.00 | 73.87 | 88.17 | 80.39 |
| | Bengali | **85.94** | **85.85** | **85.65** | 90.35 | 81.10 | 85.48 | 81.54 | 90.60 | 85.83 |
| | Code-Mix | 73.39 | 73.36 | 73.37 | 74.42 | 75.29 | 74.85 | 72.37 | 71.43 | 71.90 |

DistilBERT-multilingual framework also provides the highest precision score of 78.00.

For Bengali, in label-specific results, the IndicBERTv2 model achieves the highest precision score of 90.35, while the multilingual BERT model-based framework attains the best recall and F1-scores of 89.76 and 86.69, respectively, for the *SUPPORTS* label. Regarding the *REFUTES* label, the IndicBERTv2 model-based framework obtains the best recall and F1-scores, while the multilingual BERT model achieves the highest precision score of 87.96.

Notably, for Code-Mix texts, the lightweight DistilBERT-multilingual model-based framework outperforms all other frameworks, achieving overall precision, recall, and F1-scores of 78.74, 78.07, and 78.15, respectively. In label-specific evaluations for the *SUPPORTS* label, the DistilBERT-multilingual model achieves the best recall and F1-scores. However, it does not achieve the highest precision; the XLM-RoBERTa model-based framework records the best recall score of 82.19. For the *REFUTES* label, the XLM-RoBERTa framework obtains the best recall and F1-scores, whereas the DistilBERT-multilingual framework achieves the best precision score. In contrast, the MuRIL model exhibits a substantial decline in performance for Code-Mix claim verification, recording a precision score of 40.26, a recall score of 44.90, and an F1-score of 36.69. Although MuRIL is trained explicitly on Indian languages, its performance significantly deteriorates when handling Hindi-English Code-Mix texts in romanized script.

Although the transformer-model-based dual

encoder framework achieved the best performance across all languages, the performance in the Smith-Waterman algorithm-based approach is quite acceptable. Without any extensive training or fine-tuning, the simple rule-based sequence alignment algorithm achieves F1-scores of 68.41, 69.38, and 70.55 for the English, Hindi, and Bengali languages, respectively. However, the performance in Code-Mix language is slightly deteriorated to 56.27. One possible reason for the low performance in Code-Mix language is its linguistic complexity, which makes it difficult to verify claims properly due to the mixing of two languages.

## 6 Conclusion

This paper introduces a novel dataset for claim verification in English and low-resourced languages, specifically Bengali, Hindi, and Hindi-English Code-Mix, leveraging large language models (LLMs) alongside a manual human annotation process. Furthermore, we developed a lightweight rule-based approach inspired by the Smith-Waterman sequence labeling algorithm and a dual-encoder claim verification framework utilizing transformer-based models. Experimental results demonstrate that the IndicBERTv2 model achieves superior performance for the low-resourced languages of Bengali and Hindi. Additionally, for English and Code-Mix languages, the XLM-RoBERTa model exhibits better performance in claim verification tasks. Additionally, the rule-based algorithm approach demonstrates an acceptable performance in English, Hindi, and Bengali languages.

However, to substantiate our findings more comprehensively and enhance the robustness of the proposed claim verification framework, developing a larger and more diverse dataset is necessary.

## 7 Limitations and Future Work

Our proposed work also has some limitations. First, for English, Hindi, and Bengali, we relied solely on online news portals, which limits our exploration of claim verification in a more diverse domain, such as social media. In our future work, we will incorporate social media data, such as Twitter or Facebook posts, to explore claim verification in social media.

Second, the dataset's size is quite limited, which restricts us from providing a more robust analysis across different languages. In our future work, we will develop the claim verification framework with a large sample-sized dataset to make a more robust framework.

Third, we fine-tuned our models with a mini-batch size of 4 and 8. We were unable to go beyond the batch size of 8 due to resource limitations. In our future work, we will experiment with a higher batch size to develop a more fine-tuned framework.

Fourth, using two separate transformer encoders for claim and evidence makes the claim verification framework a resource-intensive framework. In our future work, we will develop a single encoder-based claim verification framework that utilizes fewer resources and reduces overhead.

Lastly, the rule-based algorithmic approach for claim verification is language-dependent, with parameter (match) values that differ for each language. In our future work, we aim to develop a version that will work in languages independently, at least English, Hindi, and Bengali.

## References

1. **Alam, F., Hasan, A., Alam, T., Khan, A., Tajrin, J., Khan, N., Chowdhury, S. A. (2021).** A review of bangla natural language processing tasks and the utility of transformer models.
2. **Aly, R., Guo, Z., Schlichtkrull, M. S., Thorne, J., Vlachos, A., Christodoulopoulos, C., Cocarascu, O., Mittal, A. (2021).** The fact extraction and VERification over unstructured and structured information (FEVEROUS) shared task. **Aly, R., Christodoulopoulos, C., Cocarascu, O., Guo, Z., Mittal, A., Schlichtkrull, M., Thorne, J., Vlachos, A.**, editors, Proceedings of the Fourth

Workshop on Fact Extraction and VERification (FEVER), Association for Computational Linguistics, Dominican Republic, pp. 1–13. DOI: 10.18653/v1/2021.fever-1.1.

3. **Churina, S., Barik, A. M., Phaye, S. R. (2024).** Improving evidence retrieval on claim verification pipeline through question enrichment. **Schlichtkrull, M., Chen, Y., Whitehouse, C., Deng, Z., Akhtar, M., Aly, R., Guo, Z., Christodoulopoulos, C., Cocarascu, O., Mittal, A., Thorne, J., Vlachos, A.**, editors, Proceedings of the Seventh Fact Extraction and VERification Workshop (FEVER), Association for Computational Linguistics, Miami, Florida, USA, pp. 64–70. DOI: 10.18653/v1/2024.fever-1.6.
4. **Conneau, A., Khandelwal, K., Goyal, N., Chaudhary, V., Wenzek, G., Guzmán, F., Grave, E., Ott, M., Zettlemoyer, L., Stoyanov, V. (2020).** Unsupervised cross-lingual representation learning at scale. **Jurafsky, D., Chai, J., Schluter, N., Tetreault, J.**, editors, Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, Association for Computational Linguistics, Online, pp. 8440–8451. DOI: 10.18653/v1/2020.acl-main.747.
5. **DeepSeek-AI, Liu, A., Feng, B., Xue, B., Wang, B., et al. (2025).** Deepseek-v3 technical report.
6. **Devlin, J., Chang, M.-W., Lee, K., Toutanova, K. (2019).** Bert: Pre-training of deep bidirectional transformers for language understanding.
7. **Doddapaneni, S., Aralikatte, R., Ramesh, G., Goyal, S., Khapra, M. M., Kunchukuttan, A., Kumar, P. (2023).** Towards leaving no Indic language behind: Building monolingual corpora, benchmark and models for Indic languages. **Rogers, A., Boyd-Graber, J., Okazaki, N.**, editors, Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Association for Computational Linguistics, Toronto, Canada, pp. 12402–12426. DOI: 10.18653/v1/2023.acl-long.693.
8. **Dougrez-Lewis, J., Akhter, M. E., He, Y., Liakata, M. (2024).** Assessing the reasoning abilities of chatgpt in the context of claim verification.
9. **Grattafiori, A., Dubey, A., Jauhri, A., Pandey, A., Kadian, A., et al. (2024).** The llama 3 herd of models.
10. **Hanselowski, A., Stab, C., Schulz, C., Li, Z., Gurevych, I. (2019).** A richly annotated corpus for different tasks in automated fact-checking. **Bansal, M., Villavicencio, A.**, editors, Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL), Association for Computational Linguistics, Hong Kong, China, pp. 493–503. DOI: 10.18653/v1/K19-1046.
11. **Jiang, Y., Bordia, S., Zhong, Z., Dognin, C., Singh, M., Bansal, M. (2020).** HoVer: A dataset for many-hop fact extraction and claim verification. **Cohn, T., He, Y., Liu, Y.**, editors, Findings of the Association for Computational Linguistics: EMNLP 2020, Association for Computational Linguistics, Online, pp. 3441–3460. DOI: 10.18653/v1/2020.findings-emnlp.309.
12. **Khanuja, S., Bansal, D., Mehtani, S., Khosla, S., Dey, A., Gopalan, B., Margam, D. K., Aggarwal, P., Nagipogu, R. T., Dave, S., Gupta, S., Gali, S. C. B., Subramanian, V., Talukdar, P. (2021).** Muril: Multilingual representations for indian languages.
13. **Nayak, R., Joshi, R. (2022).** L3cube-hingcorpus and hingbert: A code mixed hindi-english dataset and bert language models. arXiv preprint arXiv:2204.08398.
14. **OpenAI (2023).** Gpt-3.5 turbo. https://platform.openai.com/docs/models/gpt-3.5-turbo.
15. **OpenAI (2024).** Gpt-4o mini. https://platform.openai.com/docs/models/gpt-4o-mini.

16. **Pan, L., Chen, W., Xiong, W., Kan, M.-Y., Wang, W. Y. (2021).** Zero-shot fact verification by claim generation. **Zong, C., Xia, F., Li, W., Navigli, R.**, editors, Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers), Association for Computational Linguistics, Online, pp. 476–483. DOI: 10.18653/v1/2021.acl-short.61.

17. **Pankovska, E., Schulz, K., Rehm, G. (2022).** Suspicious sentence detection and claim verification in the covid-19 domain. **Petrocchi, M., Viviani, M.**, editors, Advances in Information Retrieval: 44th European Conference on IR Research, ECIR 2022, Proceedings, Part II. Workshop on Reducing Online Misinformation Through Credible Information Retrieval (ROMCIR-2022), befindet sich European Conference on Information Retrieval (ECIR) 2022, April 1-1, Stavanger, Norway, University of Stavanger, CEUR-WS, University of Stavanger Kjell Arholms gate 41 4021 Stavanger Norway, Vol. 13186.

18. **Pradeep, R., Ma, X., Nogueira, R., Lin, J. (2020).** Scientific claim verification with vert5erini.

19. **Saakyan, A., Chakrabarty, T., Muresan, S. (2021).** COVID-fact: Fact extraction and verification of real-world claims on COVID-19 pandemic. **Zong, C., Xia, F., Li, W., Navigli, R.**, editors, Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), Association for Computational Linguistics, Online, pp. 2116–2129. DOI: 10.18653/v1/2021.acl-long.165.

20. **Sanh, V., Debut, L., Chaumond, J., Wolf, T. (2019).** Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. ArXiv, Vol. abs/1910.01108.

21. **Sarrouti, M., Ben Abacha, A., Mrabet, Y., Demner-Fushman, D. (2021).** Evidence-based fact-checking of health-related claims. **Moens, M.-F., Huang, X., Specia, L., Yih, S. W.-t.**, editors, Findings of the Association for Computational Linguistics: EMNLP 2021, Association for Computational Linguistics, Punta Cana, Dominican Republic, pp. 3499–3512. DOI: 10.18653/v1/2021.findings-emnlp.297.

22. **Schlichtkrull, M., Guo, Z., Vlachos, A. (2023).** Averitec: A dataset for real-world claim verification with evidence from the web.

23. **Smith, T. (1981).** Smith-waterman algorithm. Advances in Applied Mathematics, Vol. 2, pp. 482–489.

24. **Sundriyal, M., Malhotra, G., Akhtar, M. S., Sengupta, S., Fano, A., Chakraborty, T. (2022).** Document retrieval and claim verification to mitigate COVID-19 misinformation. **Chakraborty, T., Akhtar, M. S., Shu, K., Bernard, H. R., Liakata, M., Nakov, P., Srivastava, A.**, editors, Proceedings of the Workshop on Combating Online Hostile Posts in Regional Languages during Emergency Situations, Association for Computational Linguistics, Dublin, Ireland, pp. 66–74. DOI: 10.18653/v1/2022.constraint-1.8.

25. **Thorne, J., Vlachos, A., Christodoulopoulos, C., Mittal, A. (2018).** FEVER: a large-scale dataset for fact extraction and VERification. **Walker, M., Ji, H., Stent, A.**, editors, Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers), Association for Computational Linguistics, New Orleans, Louisiana, pp. 809–819. DOI: 10.18653/v1/N18-1074.

26. **Wadden, D., Lin, S., Lo, K., Wang, L. L., van Zuylen, M., Cohan, A., Hajishirzi, H. (2020).** Fact or fiction: Verifying scientific claims. **Webber, B., Cohn, T., He, Y., Liu, Y.**, editors, Proceedings

of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), Association for Computational Linguistics, Online, pp. 7534–7550. DOI: 10.18653/v1/2020.emnlp-main.609.

27. **Wadden, D., Lo, K. (2021).** Overview and insights from the SCIVER shared task on scientific claim verification. **Beltagy, I., Cohan, A., Feigenblat, G., Freitag, D., Ghosal, T., Hall, K., Herrmannova, D., Knoth, P., Lo, K., Mayr, P., Patton, R. M., Shmueli-Scheuer, M., de Waard, A., Wang, K., Wang, L. L.**, editors, Proceedings of the Second Workshop on Scholarly Document Processing, Association for Computational Linguistics, Online, pp. 124–129.

28. **Wadden, D., Lo, K., Kuehl, B., Cohan, A., Beltagy, I., Wang, L. L., Hajishirzi, H. (2022).** SciFact-open: Towards open-domain scientific claim verification. **Goldberg, Y., Kozareva, Z., Zhang, Y.**, editors, Findings of the Association for Computational Linguistics: EMNLP 2022, Association for Computational Linguistics, Abu Dhabi, United Arab Emirates, pp. 4719–4734. DOI: 10.18653/v1/2022.findings-emnlp.347.

29. **Wadden, D., Lo, K., Wang, L. L., Cohan, A., Beltagy, I., Hajishirzi, H. (2022).** MultiVerS: Improving scientific claim verification with weak supervision and full-document context. **Carpuat, M., de Marneffe, M.-C., Meza Ruiz, I. V.**, editors, Findings of the Association for Computational Linguistics: NAACL 2022, Association for Computational Linguistics, Seattle, United States, pp. 61–76. DOI: 10.18653/v1/2022.findings-naacl.6.

30. **Wang, H., Shu, K. (2023).** Explainable claim verification via knowledge-grounded reasoning with large language models.

31. **Wuehrl, A., Menchaca Resendiz, Y., Grimminger, L., Klinger, R. (2024).** What makes medical claims (un)verifiable? analyzing entity and relation properties for fact verification. **Graham, Y., Purver, M.**, editors, Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers), Association for Computational Linguistics, St. Julian's, Malta, pp. 2046–2058.

32. **Yang, A., Yang, B., Hui, B., Zheng, B., Yu, B., et al. (2024).** Qwen2 technical report.

33. **Zhang, Z., Li, J., Fukumoto, F., Ye, Y. (2021).** Abstract, rationale, stance: A joint model for scientific claim verification. **Moens, M.-F., Huang, X., Specia, L., Yih, S. W.-t.**, editors, Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Online and Punta Cana, Dominican Republic, pp. 3580–3586. DOI: 10.18653/v1/2021.emnlp-main.290.

*Article received on 30/04/2025; accepted on 05/09/2025.*
**Corresponding author is Pritam Pal.*