Neural Computing and Applications (2023) 35:11337–11357
https://doi.org/10.1007/s00521-023-08300-x

ORIGINAL ARTICLE

Towards automated check-worthy sentence detection using Gated Recurrent Unit

Ria Jha[1] · Ena Motwani[1] · Nivedita Singhal[1] · Rishabh Kaushal[1]

Received: 31 July 2021 / Accepted: 16 January 2023 / Published online: 10 February 2023

Abstract
People are exposed to a lot of information daily, which is a mix of facts, opinions, and false claims. The rate at which information is created and spread has necessitated an automated fact-checking mechanism. In this work, we focus on the first step of the fact-checking system, which is to identify whether a given sentence is factual. We propose a glove embedding-based gated recurrent unit pipeline for check-worthy sentence detection, referred to as G2CW framework. It detects whether a given sentence has check-worthy content in it or not; furthermore, if it has check-worthy content, whether it is important or not, from a fact-checking perspective. We evaluate our proposed framework on two datasets: a standard ClaimBuster dataset commonly used by the research community for this problem and a self-curated IndianClaim dataset. Our G2CW framework outperforms prior work with 0.92 as F1-score. Furthermore, our G2CW framework, when trained on the ClaimBuster dataset, performs the best on the IndianClaims dataset.

Keywords Fact checking · Deep learning · Sentence classification

1 Introduction

In today's day and age, an unprecedented amount of information is constantly generated ([5, 15]). The information available to the public formulates public opinion and understanding of current events ([18, 27]). It is of utmost importance, especially in a democratic nation, that people have access to accurate information which formulates public opinion. We also observe that this information comprises facts, misleading statements, and false claims ([12]). Hence, distinction must be made between truthful, factual pieces of information and fabricated ones. Fact-checking ([6, 7, 10]) is the key to ensure transparency and accountability of those in power. Fact-checkers and journalists constantly work to identify check-worthy statements, verify the facts, and correct misinformation before making it available to the public. It is essential to automatically ([17, 25, 26]) distinguish between facts that are check-worthy, facts that do not require verification, and statements that are not factual. If the entire corpus of information is considered for verification, one will waste resources on sentences that do not warrant verification. Therefore, detecting check-worthy sentences in the first step helps to reduce the volume of information to be verified.

In this work, our focus lies on *check-worthy sentence detection*. The goal is to find whether a sentence is factual or not and whether the sentence, if factual, is worthy of verification or not. The manual process is intellectually demanding, time-consuming, and subjective to human bias. These challenges have prompted the creation and development of automated fact-detecting and fact-checking systems. As depicted in Table 1, there can be three categories of sentences. The first category comprises sentences that are not factual. The second category is those sentences that are factual, important, and worthy of further checks. Finally, the third category of sentences is factual but not

✉ Rishabh Kaushal
rishabhkaushal@igdtuw.ac.in

Ria Jha
riajha003btit17@igdtuw.ac.in

Ena Motwani
ena040btece17@igdtuw.ac.in

Nivedita Singhal
nivedita027btit17@igdtuw.ac.in

[1] Department of Information Technology, Indira Gandhi Delhi Technical University for Women, Delhi, India

Table 1 Examples of three types of sentences

Category	Illustrative sentences
Sentences that are not factual	Let me help the governor
	I really don't think it was a workable idea
	But, I'll suggest you do what you want
Factual sentences that are check-worthy	The industry of natural gas and oil contributes $4 billion a year in corporate welfare initiatives
	The proof of that lies in the fact 1 out 6 people are under the poverty line
Factual sentences but not important	You know I saw Crocodile Dundee
	I was in Houston yesterday meeting a group of hard working citizens
	Just yesterday I was shaking some hands in Toledo

important. The primary aim of our work is to detect factual statements or sentences that are truly worthy of verification, referred to as, *check-worthy* sentence. Detection of such sentences would help reduce the volume to be processed by fact-checking systems and fulfill the first step toward automated fact detection and verification systems.

Previous works have been done on the problem of check-worthiness by [4], who curated datasets, namely CT-CWC-18 and CT-CWC-19, for identification and verification of political claims. This work created a list of sentences ordered by their worthiness for fact-checking. However, their corpus is limited in terms of volume. In contrast, [1] have curated a large ClaimBuster dataset which comprises 23,533 claims from presidential debates held in the USA from 1960 to 2016. Previously, work has been done on this dataset by [12], which has explored classical machine learning models for detecting check-worthy claims, with support vector machines (SVMs) giving the best F1-score of 0.81 on the ClaimBuster dataset. In this work, we go beyond the conventional machine learning algorithms in the direction of deep neural network models to make a distinction between the three classes of sentences, namely non-factual sentence (NFS), unimportant factual sentence (UFS), and check-worthy factual sentence (CFS), automating the check-worthy fact detection process.

We propose *G2CW framework* which is based on glove embedding and gated recurrent unit (GRU) for check-worthy fact detection. Glove embeddings capture word similarities, and GRUs take into consideration long-term dependencies. Our proposed framework outperforms the previous best F1-score of 0.81 by [13] and increases the F1-score to 0.92 using the least amount of training time. Furthermore, we curated a new dataset, referred to as the IndianClaims[1] dataset. It comprises 953 claims collected from three sources: Question-Hour debates of the Indian parliament, tweets posted by politicians, and Prime Minister statements. Our G2CW framework, when trained using the ClaimBuster dataset, gives an F1-score of 0.70 when tested on the IndianClaims dataset. To summarize, our work pushes state of the art to solve check-worthiness by applying deep neural network models.

- We propose a glove embedding and GRU-based G2CW framework, which achieves an F1-score of 0.92, outperforming the previous best of 0.81 proposed by [12].
- We curated a new dataset named the IndianClaims dataset comprising 953 sentences taken from Question-Hour debates in the Indian parliament, tweets from Indian politicians, and Prime Minister statements.
- Our proposed G2CW framework trained on the ClaimBuster dataset achieves an F1-score of 0.70 on the IndianClaims dataset, which is a good starting point for check-worthy sentence detection in the Indian context.

The paper is organized as follows. This introduction section discusses the problem statement, motivation, approach, and brief results. In the next section, we discuss related work that has been done on various datasets for check-worthy sentence detection. Next, we present the dataset description section, which details the dataset attributes and their description. The proposed methodology section explains our approach toward the check-worthiness problem and describes our proposed G2CW framework. The section on results provides the outputs obtained from the various experiments done during this work. Finally, in the conclusion and future work section, we describe the summary and future scope of the work.

[1] Available on request from the corresponding author.

2 Related work

In this section, we explain the works related to fact-checking in general and check-worthiness in particular. We divide this section into two parts. In the first subsection, we discuss the works that solve the problems of fact-checking. And then, in the following subsection, we focus on prior works that address check-worthiness detection.

2.1 Fact-checking

This subsection focuses on prior works that will help fact-check mechanisms. [10] proposed a fact-checking system called ClaimBuster, using a three-class classification and ranking algorithm. Using supervised learning algorithms, the system classified the sentences into non-factual, unimportant, and check-worthy. The methods used were naive Bayes, random forest, and SVM, which were further processed by fourfold cross-validation. For the experiment, SVM had the best accuracy with various combinations of the extracted features like the parts-of-speech (POS) tag. ClaimBuster received 79% precision and 74% recall for check-worthy statements. The authors also concluded that the models had better accuracy on non-factual and check-worthy sentences than unimportant sentences. [22] introduced a fact-checking system called FAKTA, which is a composition of document retrieval from various reliable sources, stance detection of documents concerning given claims, evidence extraction, and linguistic analysis. For the experiment, the Fact Extraction and Verification (FEVER) dataset[2] was used, where sentences are labeled as refuted (REF), supported (SUP), and not enough information(NEI). The author concluded that FAKTA can predict the factuality of the given claims and provide evidence to support its prediction at the document level. [3] proposed a method to generate veracity justifications for the fact-checking system's predictions. The dataset used for the experiment was Politifact[3] and the models used were based on DistilBERT, with the macro F1-score for validation as 0.321 and 0.443, and for testing, the scores are 0.323 and 0.443. [19] proposed a web-based framework for fact-checking over Whatsapp[4]. The tool monitors two public WhatsApp groups discussing political topics: India and Brazil, in various content categories. The author also concluded that this tool would help the fact-checkers verify the information from various media sources. [21] proposed a fact-checking mechanism that will generate evidence in support of the factuality of the given claim. Computing methodologies, human-centered computing, and information system concepts were used for experimenting. The author concluded that the predictions made by the platform were correct 58% of the time and 59% of the returned evidence was relevant. [25] proposed an NLP-based fact-checking system to label a claim and also provided the pieces of evidence to show the degree of truthfulness. The main aim is to analyze the given claims' veracity and decrease the human burden of manual fact-verification. [24] proposed a binary classification model to check the factuality of news and posts on COVID-19. The author curated and annotated 10,700 sentences as the dataset[5] for the experiment, including the social media posts and fake news on the pandemic. Models explored for the experiment were SVM, random forest, decision tree, and gradient boost. SVM gave the best result among the four models with an F1-score of 93.32%. [2] aimed for fact-checking and identifying check-worthy claims. The proposed model was a supervised learning model based on neural networks, SVM, and a combination of these two and the contextual and discourse features of the input. For the fact verification, justifications were generated to access the factuality of the answers in the QnA thread of the dataset[6] for fact-checking. The resultant accuracy came up as 0.635 using contextual and discourse features only. [9] analyzed the challenges faced by fact-checking systems in the countries: India, Bangladesh, and Nepal. In this work, five fact-checking organizations were interviewed from these countries to detect the challenges of fact-checkers. The work aimed to determine to what extent social media users engage with fact-checking organizations in these countries. [20] proposed a multi-classification model, which extracts information in the form of features from the answer content of the Community Question-Answering Forum to check the answer's factuality. The author also curated the dataset[7] for the experiment. The result showed a MAP value of 86.54.

2.2 Check-worthiness of sentence

This subsection focuses on prior works that detect whether a given sentence is checked worthy or not. To this end, we list such works in Table 2.

[1] proposed the ClaimBuster database with statements issued in all US national election debates and coded. The ClaimBuster database can be embedded in computer systems to identify claims that need to be verified for authenticity in various social and digital media sources. The ClaimBuster database is accessible on public platforms[8] along with explanations on the data preparation

[2] www.fever.ai.

[3] www.politifact.com.

[4] www.whatsapp-monitor.dcc.ufmg.br.

[5] https://github.com/parthpatwa/covid19-fake-news-detection.

[6] https://github.com/qcri/QLFactChecking.

[7] https://github.com/qcri/QLFactChecking.

[8] https://zenodo.org/.

Table 2 Research Papers Comparison

Paper title and author	Brief description	Methods and best results	Year
A Benchmark Dataset of Check-Worthy Factual Claims, [1]	The Work proposed a ClaimBuster database accessible with explanations of the data preparation process, descriptive data statistics, potential use cases, and various fair policies followed while creating it	work done include data collection, text cleaning, pre-processing, string parsing, and labeling sentences in three classes, namely Check Worthy Factual Sentence (CFS), Non-Factual Sentence (NFS), and Unimportant Factual Sentence (UFS)	2020
Overview of the CLEF-2018 CheckThat: Lab on Automatic Identification and Verification of Political Claims Link, [23]	The main aim is to rank the sentences on the basis of the check worthiness of the claims and to detect the level of factuality in the claim. The dataset comprises two different languages, English and Arabic	Models explored are KNN, SVM, Random Forests, Naive Bayes, Decision Trees, and Neural Network Models. The results for the English dataset are a MAP score of 0.1332 and an MAE score of 0.7050. The Arabic dataset results are a MAP score of 0.899 and an MAE score of 0.6579.	2018
CheckThat at CLEF 2019: Automatic Identification and Verification of Claims, [4]	The main aim of the work is fact-checking on the English dataset and create a ranking algorithm for sentences in the Arabic dataset	Models explored are SVM, Naive Bayes, Linear Regression, Decision Trees, and various neural network models. The MAP score for task 1 is 0.1660, nDCG score for sub-task 1 is 0.55, F1-scores for the sub-tasks 2, 3 and 4 are 0.42, 0.37, and 0.34 respectively	2019
Comparing Automated Factual Claim Detection Against Judgments of Journalism Organizations, [11]	The main aim is to identify the factual and check-worthy claims and compare its results with the judgments of other professional news organizations.	The result implied that the ClaimBuster system strongly resembled other professional organizations for fact-checking. ClaimBuster gave the highest scores (>= 0.5) on the topics chosen by organizations like CNN and PolitiFact for fact-checking	2016
Toward automated fact-checking: Detecting check-worthy factual claims by ClaimBuster, [12]	A multi-class classification system based on check-worthiness and its factuality	Models explored are Support Vector Machine (SVM), Multinomial Naive Bayes Classifier (multi-NBC), and Random Forest Classifier (RFC) and considered three combined features: Words, Words And Part of Speech Tags, Words, Part of Speech Tags and Entity type. Among all the models explored, SVM gave the best result in classification with an F1-score of 0.818	2017
Toward automated fact checking: Developing an annotation schema and benchmark for consistent automated claim detection, [16]	A binary claim classification system can identify the set of sentences are factual claims	The results of Logistic Regression and SVM are considered, and the CNC (Claim and Not Claim) model performs better than the previous ClaimBuster, with the F1-score at 0.83.	2020
Claimrank: Detecting check-worthy claims in Arabic and English, [14]	An online fact-checking system to detect check-worthy claims by prioritizing the claims to be checked first	Model gives the best result using neural networks and NLP with the MAP Score on the English dataset is 0.319 and on the Arabic dataset is 0.302	2018
ClaimBuster: the first-ever end-to-end fact-checking system, [13]	An automated fact-checking, ClaimBuster gives each sentence a score to indicate the usefulness of the claim based on fact-checking. This provides valuable assistance to fact-checkers by focusing on high-quality sentences without carefully sorting through many sentences	Approaches used are machine learning, NLP, and database query techniques	2017
Neural check-worthiness ranking with weak supervision: Finding sentences for fact-checking, [8]	A ranking algorithm to create a list of sentences based on their check-worthiness using both word embedding and syntactic dependency	The model is trained on a large unlabeled dataset using weak supervision. Then an RNN model is applied with domain-specific word embeddings and syntactic dependency parsing of a string. The best result is a MAP score of 0.278	2019
A Hybrid Recognition System for Check-worthy Claims Using Heuristics and Supervised Learning, [28]	A hybrid approach to identify factual claims based on its check-worthiness by combining simple heuristics with supervised ML algorithms and further create a rank-list of the set of factual and check-worthy claims	For the experiment, two supervised learning algorithms, MultiLayer Perceptrons (MLP) and Support Vector Machines (SVM), and an ensemble model (that combines SVM and MLP) are used. The result is a MAP score of 0.1332	2018

process, descriptive data statistics, potential use cases, and various fair policies followed while creating it. Another important work was to label the collected and processed data into three classes, namely check-worthy factual sentence (CFS), non-factual sentence (NFS), and unimportant factual sentence (UFS). [23] proposed a model to accomplish two primary goals: check-worthiness and factuality. The first goal is to create a rank list of all the potential claims made during the speeches and debates based on their check-worthiness. The second goal is to detect whether the claim is likely true, half-true, or false. The dataset comprises both English and Arabic languages[9]. The evaluation metric for the first task is mean average precision (MAP), and for the second task, it is mean absolute error (MAE). Models tested for these tasks are KNN, SVM, random forests, naive Bayes, decision trees, and various neural network models. The results for the English dataset are a MAP score of 0.1332 and an MAE score of 0.7050. The results for the Arabic dataset are a MAP score of 0.899 and an MAE score of 0.6579. The author concluded that the results and performance of the models work well in Arabic compared to English, and adding annotations from different sources and increasing the corpus can work for multi-task learning. [4] proposed a model, which is the second edition of CheckThat! Lab at CLEF 2019. The model aimed at two main tasks for the languages, Arabic and English, in the dataset[10]. The first task is done on the English dataset, which detects whether the claims made during speeches and debates should be prioritized for fact-checking. The second task is done on the Arabic dataset; this task is divided further into four sub-tasks. The first sub-task is to create a rank list of web pages based on their check-worthiness of the claims. The second sub-task is the classification of the webpages from the first sub-task on the basis of their usefulness in fact-checking the potential claim. The third sub-task is to extract the useful text passages from the web pages in the second sub-task. The fourth sub-task is to detect the check-worthy claim from useful passages of the third sub-task. Features used are parts-of-speech (PoS) tags, bag-of-words (BOW) representations, sentiment analysis, named entities (NEs), and many more. SVM, naive Bayes, linear regression, decision trees, and various neural network models are explored. The evaluation metrics for the model are the mean average precision (MAP) score, and normalized discounted cumulative gain (nDCG) score for ranking, and F1-score for classification. The MAP score for task 1 is 0.1660, nDCG score for sub-task 1 is 0.55, F1-scores for the sub-tasks 2, 3 and 4 are 0.42, 0.37, and 0.34 respectively. [11] proposed an automated fact-checking system and compared its results with the judgments of other professional news organizations. The main aim is to identify factual and check-worthy claims. Results are compared with CNN[11] and PolitiFact[12]. Topic detection is performed, and ClaimBuster gave the highest score (greater than 0.5) on the topics chosen by organizations like CNN and PolitiFact for fact-checking. The author concluded that the distribution of fact-worthy claims is studied across parties, candidates, and topics. Currently, fully automatic methods for fact-checking still fall short in terms of quality, and hence credibility and hence final confirmation by humans are still deemed necessary. [12] proposed a model for multi-class classification using the models support vector machine (SVM), multinomial naive Bayes classifier (multi-NBC), and random forest classifier (RFC) and considered three combined features: Words, Words And part-of-speech tags, Words, Part of Speech Tags and Entity type. The evaluation metric for the classification was the F1-score, and for ranking, it was Precision-at-k (P@k), AvgP (Average Precision), and nDCG (Normalized Discounted Cumulative Gain). Among all the models explored, SVM gave the best result in classification with an F1-score of 0.818. [16] proposed a model which is named as claim and not Claim. Logistic regression and SVM results are considered, and the CNC model performs better than the ClaimBuster, with the F1-score at 0.83. The author concluded that F1 results are of binary claim, but better results can be produced using classification as in the annotation schema. Further collaborating with other fact-checking organizations and including other languages, the author also concluded that the corpus could enhance and collect more data from other sources, such as social media and print outlets. [14] proposed a neural network model with two hidden layers. Features given as input contain information about the claim and the context. First Hidden Layer- Number of neurons: 200 and Activation Function: ReLU. Second Hidden Layer- Number of neurons: 50 and Activation Function: ReLU. Output Layer- Activation Function: Sigmoid. The output unit classifies the given input as check-worthy or not. The model gives the best result using neural networks and NLP with the MAP Score on the English dataset is 0.319 and on the Arabic dataset is 0.302. The author concluded that the dataset is limited in genre and language, so it needs to be expanded by considering the political debates and speeches on other genres and multiple Languages to be considered in the future. [13] proposed an automated fact-checking system using machine learning, NLP, and database query techniques. The system monitored live political/general discussions (interviews, speeches, and debates), news, and social media to find check-worthy factual

[9] https://github.com/clef2018-factchecking/clef2018-factchecking.

[10] https://sites.google.com/view/clef2019-checkthat/datasets-tools.

[11] https://edition.cnn.com/.

[12] https://www.politifact.com/

claims. The ClaimBuster[13] system architecture comprises of claim monitor, claim spotter, claim matcher, claim checker, and fact-check reporter. ClaimBuster gives each sentence a score to indicate the usefulness of the claim based on fact-checking. This provides valuable assistance to fact-checkers by focusing on high-quality sentences without carefully sorting through many sentences. We conclude that the research is limited to regional languages and political periods only. [8] proposed a deep learning model for ranking the sentences based on their check-worthiness using both word embedding and syntactic dependency. Word embedding aimed to extract semantic features from the sentences, and syntactic dependency parsing aimed to extract the role of the word in changing the semantics of other words in the sentence. The model is trained on a large unlabeled dataset (documents related to US Presidency Project[14]) using weak supervision and then an RNN model applied with domain-specific word embeddings and syntactic dependency parsing of a string. The evaluation parameter considered by the author for the ranking was mean average precision(MAP). The result was a MAP score of 0.278. According to the author, future work involves researching various symbols and inserting the context of the text's meaning into a model. [28] proposed a hybrid approach to identify factual claims based on their check-worthiness by combining simple heuristics with supervised ML algorithms and further creating a rank-list of the set of factual and check-worthy claims. The main aim of the model is to rank the statements and automatically detect whether the claims are worth checking. For the experiment, two supervised learning algorithms, multilayer perceptrons (MLP) and support vector machines (SVM), and an ensemble model (that combines SVM and MLP) is used. The evaluation metric for the experiment is mean average precision (MAP), and the result is a MAP score of 0.1332. The author concluded that the model proposed has not been used earlier for testing check-worthy sentences.

After studying prior work, we find that the prior research on the Claimbuster dataset has trained the models on the partial dataset. In contrast, our proposed G2CW Framework trains the model on the complete dataset. The previous research paper does not experiment with state-of-the-art deep learning models and has only trained the dataset on the baseline models (naive Bayes, SVM, random forest). In contrast, our G2CW Framework is based on GRU leverage word-level dependencies to detect check-worthy sentences. The research paper also explores various other neural network models.

[13] https://idir.uta.edu/ClaimBuster.

[14] https://web.archive.org/web/20170606011755/http://www.presidency.ucsb.edu/.

3 Dataset description and analysis

3.1 Description of datasets

ClaimBuster Dataset: We use the ClaimBuster dataset ([1]) which comprises 23,533 sentences, and every statement is classified into check-worthy factual claim (CFS), non-factual claim (NFS), and unimportant factual claim (UFS). The data is developed using the statements provided by the presidential members during the past presidential election debates. Almost 101 programmers had labeled these statements for 26 months in various stages. The data is organized in the following three files:- (1) *groundtruth* file: It contains only testing sentences whose labels were settled upon by three specialists. (2) *crowdsourced* file: It comprises sentences that experts labeled. (3) *allsentences* file: It contains both the ground truth and crowd-sourced sentences, and the label is missing. Table 3 describes the attributes in the *groundtruth* and *crowdsourced* files.

IndianClaims dataset: We curated a dataset using data from the Indian political context. India is the largest democracy in the world and follows a parliamentary form of governance. The parliament comprises elected representatives. We built an Indian dataset comprising 953 statements collected from three different sources: (i) Question-Hour[15]—500 sentences from the Lok Sabha Dataset, (ii) Tweets[16]—350 sentences from the tweets given by the political leaders of India, and (iii) PM statements[17]—103 sentences issued by the Prime Minister of India during debates. Keeping our Indian dataset consistent with the ClaimBuster dataset, we categorize each statement as a non-factual claim (NFC), unimportant factual claim (UFC), or important factual claim (CFS). Question-Hour is the dataset of the Parliamentary Lok Sabha[18] that includes both the questions and answers being asked in the Lok Sabha by the various members of different parties on different dates on different ministry topics. We select tweets posted by Indian political leaders and ministers holding key portfolios: Narendra Modi, Amit Shah, Piyush Goyal, Rahul Gandhi, and Nirmala Sitharaman. PM statements comprise the debates given by the Prime Minister of India on different topics, and among these debates, we select

[15] In the Indian Parliament, the Question Hour refers to the time allocated in which the elected representatives ask questions about different domains, and the associated ministers are expected to reply very specifically, http://loksabhaph.nic.in/Questions/questionlist.aspx.

[16] Tweets posted by well-known politicians in India since the COVID-19 pandemic began, https://kaggle.com/ajiteshshukla98/tweet-data.

[17] Official statements issued by Prime Minister of India, https://mea.gov.in/.

[18] TPCD-IPD: TCPD Indian Parliament Dataset 1.0". Trivedi Centre for Political Data, Ashoka University.

Table 3 Attributes of a standard benchmark ClaimBuster dataset ([1])

Attributes	Description
Sentence_id	A unique integral identifier to distinguish sentences in the dataset
Text	A sentence that a debate member delivered
Speaker	Name of the person who was delivering the sentence
Speaker_title	Speaker's designation at the time of the debate
Speaker_party	The political affiliation of the speaker
File_id	Debate record identifier
Length	The number of words in the sentence
Line_number	An integral identifier to signify the order of the sentences according to the debate transcript
Sentiment	Score represents the sentiment, which ranges from -1 to $+1$, where -1 represents the most negative sentiment, and $+1$ represents the most positive sentiment
Verdict	Labels assigned to the sentences: for CFS, it is 1, for UFS, it is 0, and for NFS it is -1

only a few sentences. Table 4 describes the attributes in the dataset.

3.2 Data analysis

3.2.1 ClaimBuster dataset

The dataset contains the sentences spoken during 33 US presidential debates from 1960 to 2016. Figure 1 represents the variation of claims over the years in the debate. The X-axis represents all the years of debates, and Y-axis represents the count of sentences based on each class of claim, namely NFS, CFS, and UFS.

This plot represents the number of sentences based on each class over the 33 election debates from 1960 to 2016. Also, we can see that the non-factual statements are most occurring. Moreover, the amount of check-worthy sentences is most common among the factual claims. We also analyzed the distribution over the total count of sentences and the average length of sentences per debate each day in Figs 2 and 3, respectively. We observed that the count of sentences has increased by approximately 60% while the length of speech has decreased by nearly 47%. We also observed that the nature of curves in both figures is inversely proportional, which indicates that by the years 1960-2016, the sentences per debate each day increased while its length decreased.

There were a total of 69 presidential members in the general election debates. These candidates were part of three political parties: Republican, Democrat, and Independent Party. The division of these members was such that 33 out of 69 were from Republican Party, 32 were from the Democrat party, and the rest 4 were Independent candidates.

Figure 4 represents the distribution of sentences type among the speakers from different political parties. We observe that the democrats had the highest number of check-worthy claims while the republicans had the highest number of non-factual claims.

Figure 5 represents the positive sentiment score frequency for check-worthy and factual claims given by republicans and democrats, respectively. A positive sentiment score was taken to check how many parties gave positive and factual claims during speeches and debates. We infer that the positive sentiment score frequency curve for the democrat and republican parties behaves similarly. There were 20 speakers in the 33 debates, and ten designations of the speakers were presented in the dataset.

Figure 6 represents the distribution of claims over the speaker's designation. It can be inferred that the Governor gave the most non-factual claims, and the President made the most check-worthy and factual claims.

Figure 7 represents the claim distribution over speakers in the elections. George Bush contributed to the highest non-factual claims among the speakers, and Barack Obama contributed to the highest check-worthy factual claims.

3.2.2 IndianClaims dataset

In this section, we perform an analysis of the Indian dataset. Recall that we annotated each statement into the same three categories defined earlier for ClaimBuster Dataset, namely NFS (non-factual statements), UFS (unimportant factual statements), and CFS (check-worthy factual statements). Figure 8 represents the count of sentences per class. It is evident from the plot that non-factual statements have the maximum count, while among the factual statements, check-worthy sentences are more.

Table 4 Attributes of the self-curated IndianClaims dataset

Attributes	Description
Text	The question raised answered content, tweets and statements reported on the Lok Sabha official site and in the tweet database.
Verdict	Labels assigned to the sentences: for CFS, it is 1, for UFS, it is 0, and for NFS it is −1

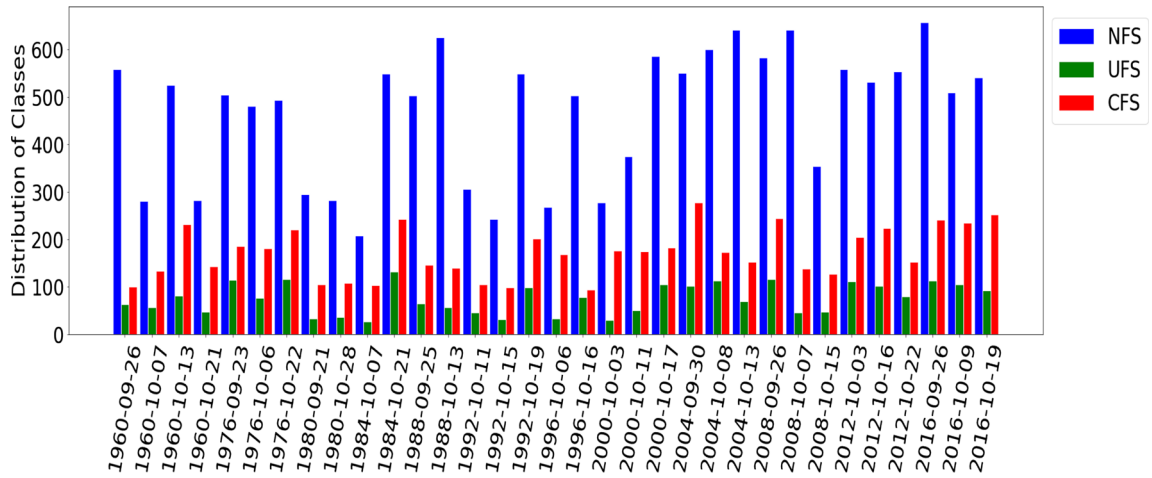

Fig. 1 Distribution of sentences per debate in 1960-2016

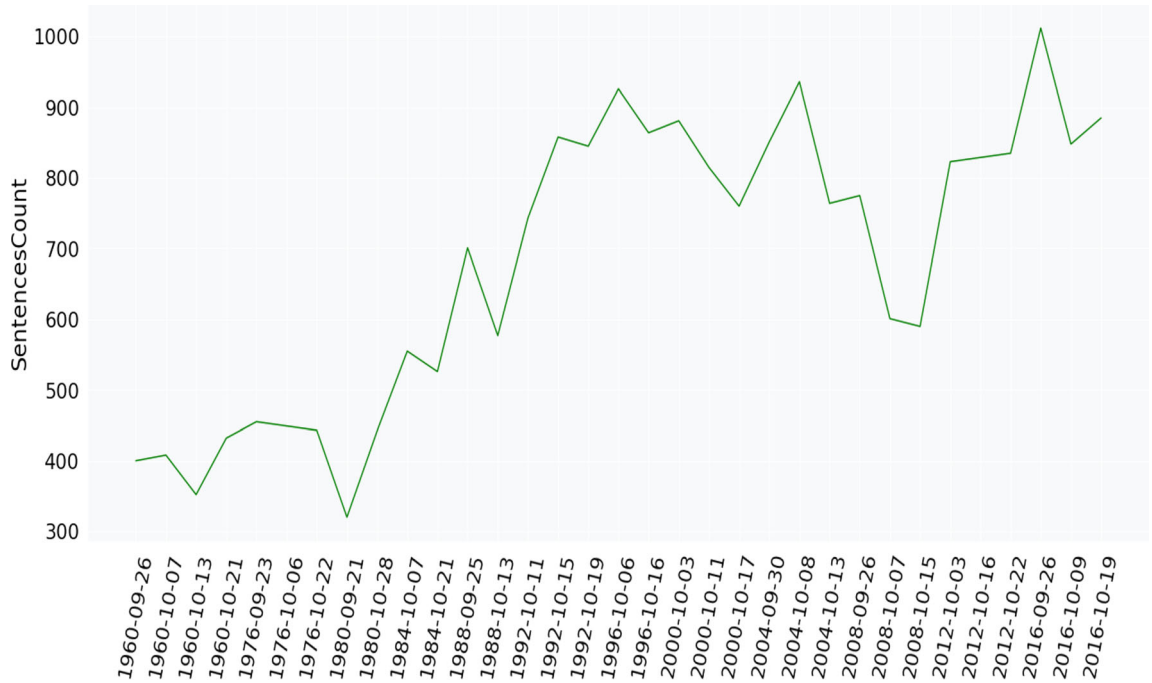

Fig. 2 Distribution of total sentences counts per debate each day

Figure 9 refers to the word cloud made from all the sentences recorded in Indian Dataset. The word cloud consists of the 500 most frequently used words made during speeches, debates, or social media tweets in India.

Figure 10 represents the most common 30 words recorded in the dataset. It represents the mainly used words in Indian Political posts and speeches.

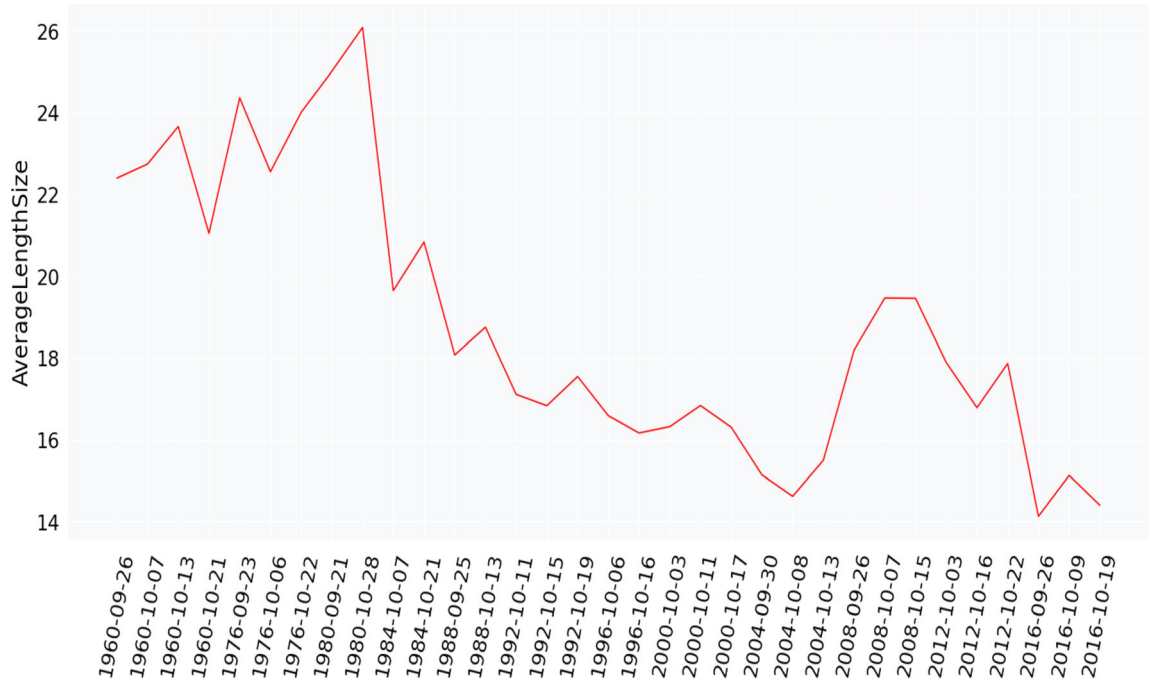

Fig. 3 Distribution of average sentence length per debate each day from 1960 to 2016

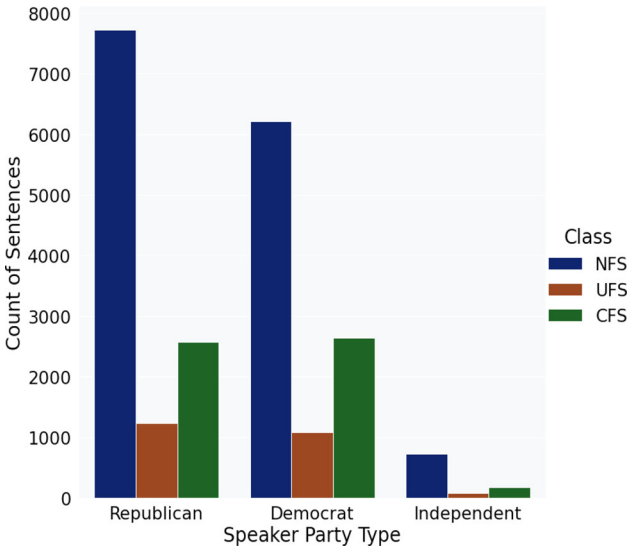

Fig. 4 Claim Distribution over Speaker Party over the years

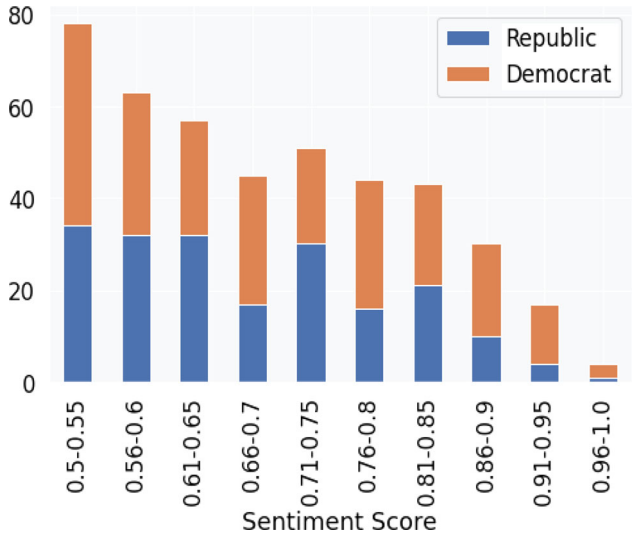

Fig. 5 Positive Sentiment Score for CFS Class of Republican and Democrat Speaker Party

4 Proposed methodology

Our proposed methodology aims to detect the check worthiness of a sentence. The main objective is twofold. First, whether a sentence is factual or not. Second, if the sentence is factual, is it worthy of verification or not? We cast this problem as a multi-class classification problem by classifying the given sentence into non-factual sentence (NFS), unimportant factual sentence (UFS), and check-worthy factual sentence (CFS). More formally, given a sentence S of words w_1, w_2, ... w_n, the goal is to learn a function F, that detects whether it is check-worthy or not.

$$F(S) = F(w_1, w_2, ..., w_n) = \begin{cases} -1, & NFS \\ 0, & UFS \\ 1, & CFS \end{cases}$$

We recall a few examples. For instance, *'Hello, Good Morning'* is a greeting and does not have any factual information. Another instance *'I woke up to the roar of a lion at my window today'* is factual but is not worth

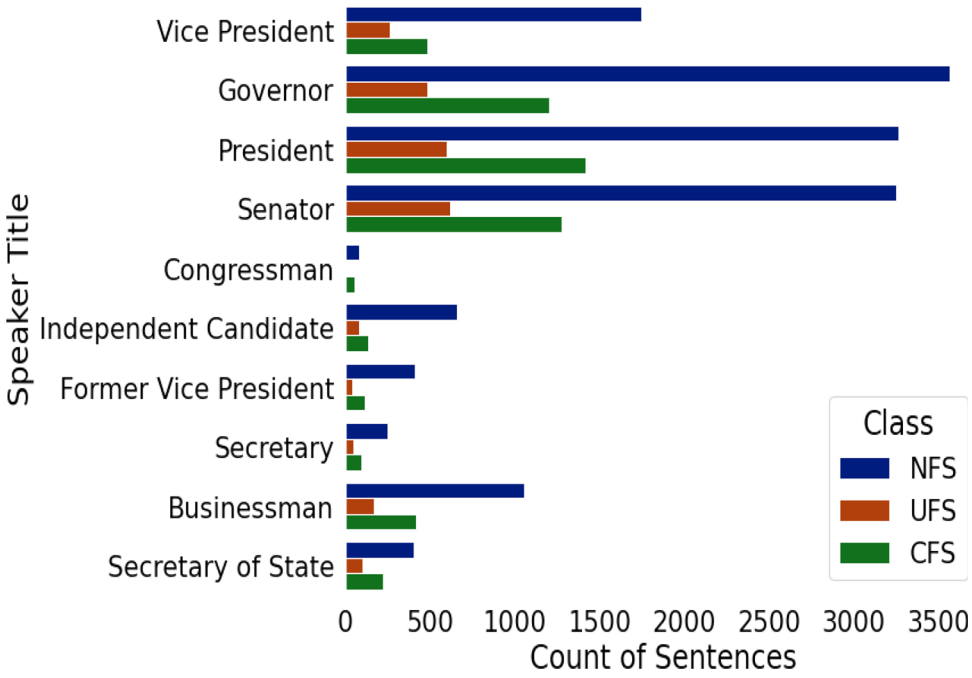

Fig. 6 Claim Distribution for Speaker Designation

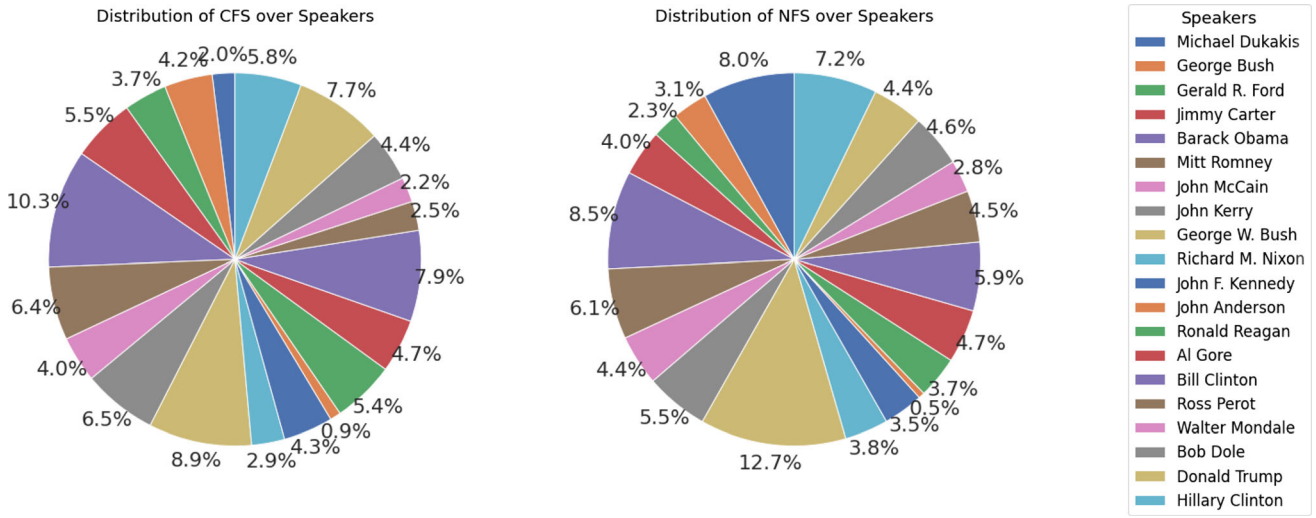

Fig. 7 Distribution of claims over Speakers

verification. However, a sentence *'India reported 32,000 Covid-19 infected cases, which is a 5% drop from December 1st'* is a factual sentence worth verifying.

4.1 Proposed G2CW framework

In this subsection, we explain our proposed approach, referred to as, *G2CW Framework*. It contains two major components, namely glove embedding and gated recurrent unit (GRU), which together will compose a deep learning model to categorize the sentences as **C**heck-**W**orthy Facts. Figure 11 describes our proposed *G2CW* framework which takes as input a sentence *'he is doing this work today'* containing 6 words. Each word is considered as the input at a given timestamp.

We denote the words in a given sentence S_i as w_i^1, w_i^2, w_i^3, ... w_i^n. In Table 5, we describe the other notations used. The words w_i^1, w_i^2, w_i^3, ... w_i^n of the sentence S_i, are passed onto the embedding layer with pre-trained weighted matrix of glove embedding to obtain vector representations of each word with the embeddings e_i^1, e_i^2, e_i^3, ... e_i^n. These word embeddings are fed as input to the gated recurrent unit (GRU) layer with N units. GRU is a type of RNN that performs sequence-to-sequence learning by considering the previous state and the current input word. After obtaining

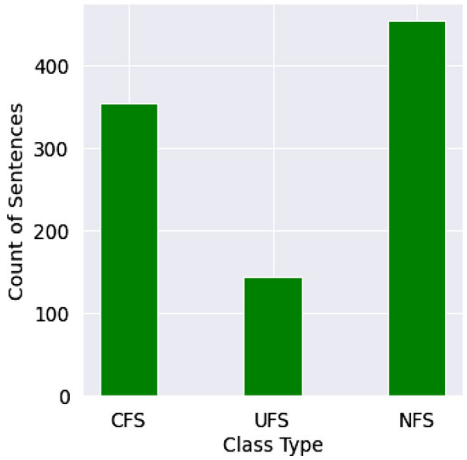

Fig. 8 Distribution of the counts of sentences in the Indian dataset over the types of the classes, namely NFS, CFS, and UFS

the output of the GRU cells, the last cell output is given to the two fully connected layers, denoted by *Dense*_1 and *Dense*_2 layers. Finally, the output layer has 3 units corresponding to the three classes (O_{cfs}, O_{nfs}, and O_{ufs}) with *softmax* as an activation function for the multi-class classification. Next, we describe each step of the *G2CW* framework in detail.

Glove embedding: Glove embedding generates embeddings of each word in the sentence and uses the global characteristics of the ClaimBuster dataset using the co-occurrence matrix.

$$J = \sum_{i,j=1}^{V} f(X_{ij}) \left[W_i^T \tilde{W}_j + b_i + b_j - ln(X_{ij}) \right]^2 \quad (1)$$

The above Eq. 1 is the cost function that is minimized for obtaining the word vectors of each word, where V refers to the vocabulary size of the ClaimBuster dataset. X_{ij} refers to the value of a particular cell in the co-occurrence matrix which is the number of occurrences of the jth word in the context of i. W_i and W_j refer to the weights, whereas b_i and b_j refers to the biases. And $f(X_{ij})$ refers to the function to improve performance.

Gated recurrent unit: A gated recurrent unit is a specialized recurrent neural network (RNN) architecture that uses the gating mechanism to solve long-term dependencies by selective reading, writing, and forgetting. GRU consists of two gates: the update gate and the reset gate. To understand the reason for our choice of GRU, consider an example statement - *Barack Obama ruled for the years 2009 - 2017 in the Democrat party, and he had lunch with his wife this evening*. In this sentence, the phrase - *'and he had lunch with his wife this evening'* is not important from a fact-worthy point of view, and therefore, it can be easily forgotten. GRU can forget such phrases by the use of forget gates. The update gate tells about the amount of information to be passed to the future. Equation 2 describes the operation of update gate.

Fig. 9 Word Cloud Analysis of the sentences recorded

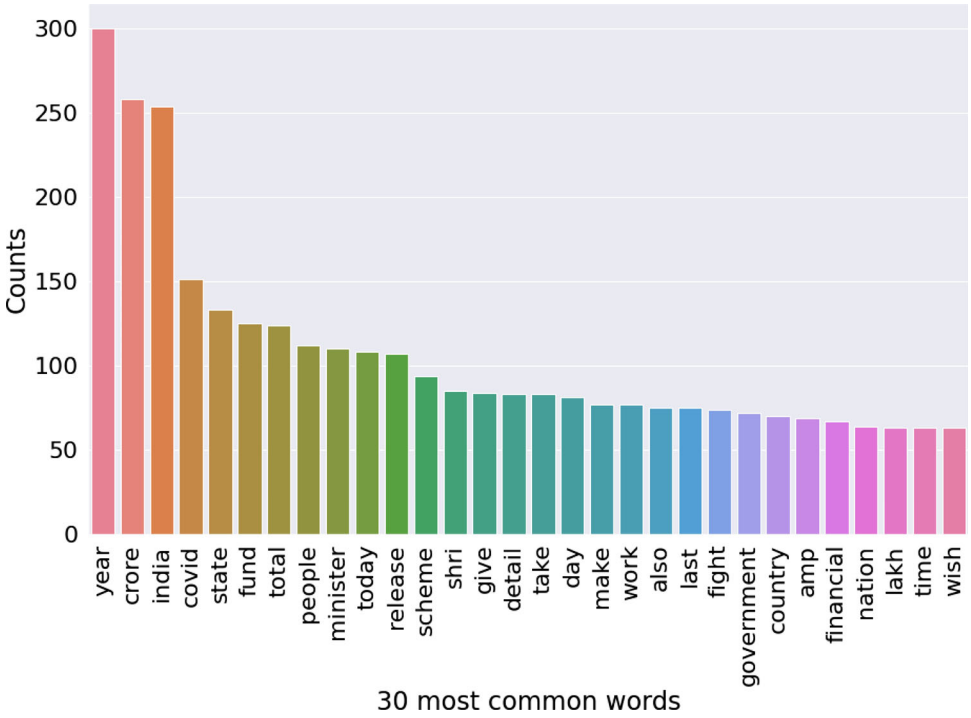

Fig. 10 Top 30 most frequent words used in the Indian Dataset

$$Z_t = \sigma(W_Z X_t + U_Z h_{t-1}) \quad (2)$$

Here, W_Z and U_Z refer to the weighted matrices. X_t refers to the current input word. Symbol σ refers to the activation function. Moreover, h_{t-1} refers to the previous states. The reset gate tells about the amount of past information to be forgotten. Equation 3 describes the operation of the reset gate.

$$R_t = \sigma(W_R X_t + U_R h_{t-1}) \quad (3)$$

Here, W_R and U_R refer to the weighted matrices. Similarly, X_t refers to the current input word, and h_{t-1} refers to the previous states. The symbol σ refers to the activation function.

Dense layer and dropout: We add two dense layers in our proposed G2CW framework, namely *Dense*_1 and *Dense*_2 followed by dropout. The dense layers are neural network layers that are fully connected. All neurons of a dense layer receive input from each of the neurons of the preceding layer. Each neuron in the dense layer receives input from all neurons of the previous layer. We use *softmax* as the activation function in the *Dense*_2 layer to perform multi-class classification of the ClaimBuster dataset. The dropout is used to avoid over-fitting the model.

In Table 6, we describe each layer's input and hyperparameters in the proposed G2CW framework. We converged on these values after performing hyperparameter tuning.

5 Experiment setup & results

5.1 Experiment design

In this section, we explain the design of our experiments. Figure 12 outlines the steps we perform, starting with data exploration and data visualization of the ClaimBuster dataset. We implement three types of word embeddings for text preprocessing, namely GLove, Word2Vec, and Keras default. In terms of machine learning models, we perform two sets of experiments. First, we replicate the baseline models, namely SVM, naive Bayes, KNN, random forest, and logistic regression [12]. Second, we implement deep learning-based approaches based on LSTM and CNN. We train all the models with different values of hyperparameters to get a final model with the best F1-score and least training time among all the best-performing models.

5.2 Results for ClaimBuster dataset

5.2.1 Baseline models

In this subsection, we explain the results of our implementations of the baseline models. We summarize our results in Table 7 by presenting average precision, recall, and F1-score values for the ClaimBuster dataset. Features *W* denotes those features we extract from the sentences spoken in the presidential debates. We preprocess the sentences before

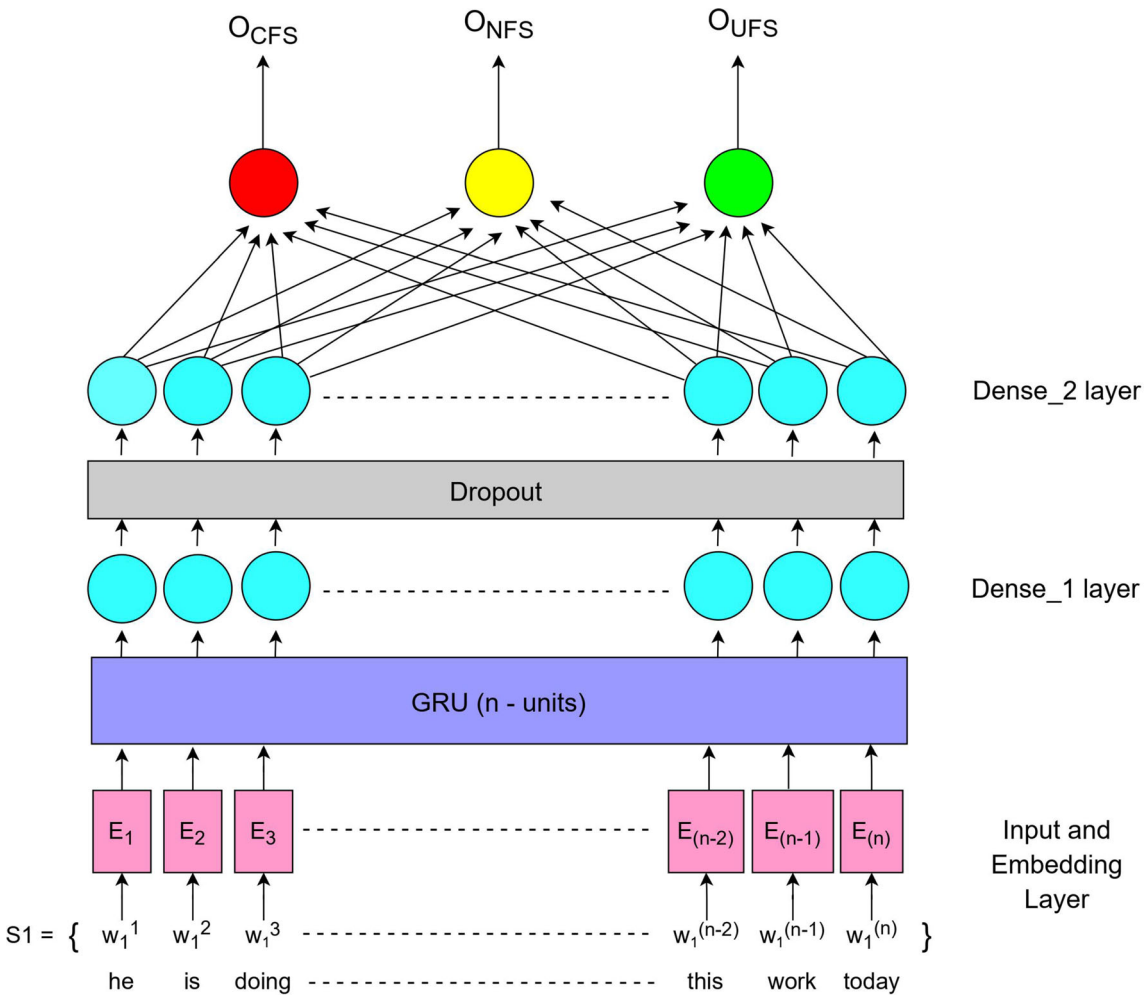

Fig. 11 Proposed G2CW framework which takes a sentence as input and outputs whether it is a non-factual sentence (NFS), unimportant factual sentence (UFS), or check-worthy factual sentence (CFS)

Table 5 Summary of Notations used in G2CW framework

Notation	Description
S_i	Refers to the ith numbered sentence
w_i^j	Represents the ith word in the S_i sentence, where j refers to the position of the word in the sentence
e_i^j	Refers to the word embedding corresponding to the jth word in ith sentence
O_{ufs}	It refers to the output as an unimportant factual sentence
O_{nfs}	It refers to the output as a non-factual sentence
O_{cfs}	It refers to the output as a check-worthy factual sentence

inputting the model by removing stop words and stemming. W_P denotes features that we extract sentences along with the Parts of Speech(POS) tags where each POS tag is considered a new feature. POS tag is important because this helps in identifying the sentences that are worthy of checking. After all, the sentences that are facts generally contain numerical values and numbers. So the sentences that need to be checked have many POS tags. It turns out that SVM with features as W_P gives the best F1-score of 0.86 among all the baseline models explored. Results of best performing models have been shown in bold in all tables.

5.2.2 Deep learning models

In this subsection, we describe the deep learning-based models. We experiment with CNN- and LSTM-based models; their brief description is given below.

- CNN_1: This model comprises 3 CNN conv_layers of 100 filters, each interleaved with 3 maxpool layers followed by 3 dense unit layers (output layer) with softmax activation function.
- CNN_4: This model contains 1 CNN conv_layers of 32 filters, each interleaved with 2 maxpool layers followed by 3 unit dense layer (output layer) with softmax activation function.

Table 6 G2CW Description and Parameters

Layer	Input parameters	Hyperparameters
Embedding layer	Weight Matrix of pretrained GloVe Embeddings	Embedding Dimension = 300
GRU	N=100 units	–
Dense_1	25 units	ReLU activation
Dense_2	3 units	Softmax activation
Dropout	0.4	–

- CNN_5: This model entails 1 CNN conv_layer of 100 filters followed by 1 maxpool layer, followed by 3 unit dense layer(output layer) with softmax activation function.
- CNN_6: This model has 1 CNN conv_layer of 32 filters followed by 1 maxpool layer, followed by 3 unit dense layer(output layer) with softmax activation function.
- LSTM_1: This model comprises 2 Layers of LSTM, with the first layer with 50 units, the second layer with 20 units, 3 units dense layer-softmax.
- LSTM_2: This model contains 1 Layer of LSTM with 100 units, 3 units dense layer-softmax.

Table 8 shows the loss and accuracy during the validation and testing phase obtained from the various deep learning models on ClaimBuster. Across all models, we observe that accuracy at the test stage is increased across all models when Word2Vec and glove embeddings are used compared to the default Keras embeddings. Among the CNN-based models, we observe that CNN_4 gives the best accuracy of 92% with Word2vec, which indicates that only one convolutional layer with 32 filters followed by two max pool layers is sufficient. Among the LSTM-based models, we find that LSTM_2 performs the best with an accuracy of 91% than LSTM_1, which is a more complex model.

Table 9 shows the class-wise precision, recall, and F1-score for the classes, namely NFS, UFS, and CFS, when we run different deep learning models on ClaimBuster Dataset. For the *NFS class*, the maximum precision of 100% is given by CNN_6 model with Default Keras as the embedding layer, CNN_1 gives the maximum recall of 56% in both the scenarios when Word2Vec and glove embedding is used, and LSTM_1 model that captures word dependencies gives the best F1-score of 66% when glove embeddings are used. For the *UFS class*, the maximum precision of 90% is given by the CNN_4 model with the glove embedding layer. The maximum recall of 92% is obtained using the LSTM_1 model with glove embedding. Moreover, the maximum F1-score of 87% is achieved by using CNN_5 and LSTM_1 models with glove embeddings. For the *CFS class*, the maximum precision of 96% is obtained using LSTM_1 and LSTM_2 models with glove embeddings. CNN_4 and CNN_6 models give a recall of 98% with glove embeddings, and the CNN_4 model also gives a maximum recall of 98% with the Default Keras embedding layer. Lastly, the LSTM_2 model gives the best F1-score of 96% with glove embeddings.

Table 10 shows the weighted average values of precision, recall, and F1-score from the various deep learning models on ClaimBuster that we have explored. The weighted average was taken to compensate for the data imbalance problem, and weights are proportional to the number of records of each class. The LSTM_2 model gives the maximum weighted average precision of 92% with glove embedding. Many models give the best value of weighted average recall of 91%, namely CNN_1 and CNN_4 with Word2Vec embedding, and CNN_1, CNN_5, LSTM_1, and LSTM_2 with glove embeddings. Similarly, we achieve a maximum weighted average F1-score of 91% using CNN_1 and CNN_4 with Word2Vec embedding, and CNN_1, CNN_5, LSTM_1, and LSTM_2 with glove embedding.

5.2.3 Best-performing models

In addition to the deep learning models discussed so far, in this section, we explain those deep learning models that gave us the best results on the ClaimBuster dataset. We refer to them as best-performing models on ClaimBuster Dataset, and they are enlisted as follows.

- CNN_2: This model contains 3 CNN conv_layers of 32 filters, each interleaved with 3 max pool layers followed by 3 unit dense layers (output layer) with softmax as the activation function.
- CNN_3: It comprises 2 CNN conv_layers of 100 filters, each interleaved with 2 max pool layers followed by 3 dense unit layers (output layer) with softmax activation function.
- LSTM_3: This model has 1 layer of Bidirectional LSTM with 100 units, 3 units of dense layer followed by softmax.
- LSTM_4: This model contains 1 layer of LSTM with 32 units, 3 units dense layer followed by softmax.
- LSTM_5: This model comprises of 1 layer of Bidirectional LSTM with 32 units and 3 units dense layer.

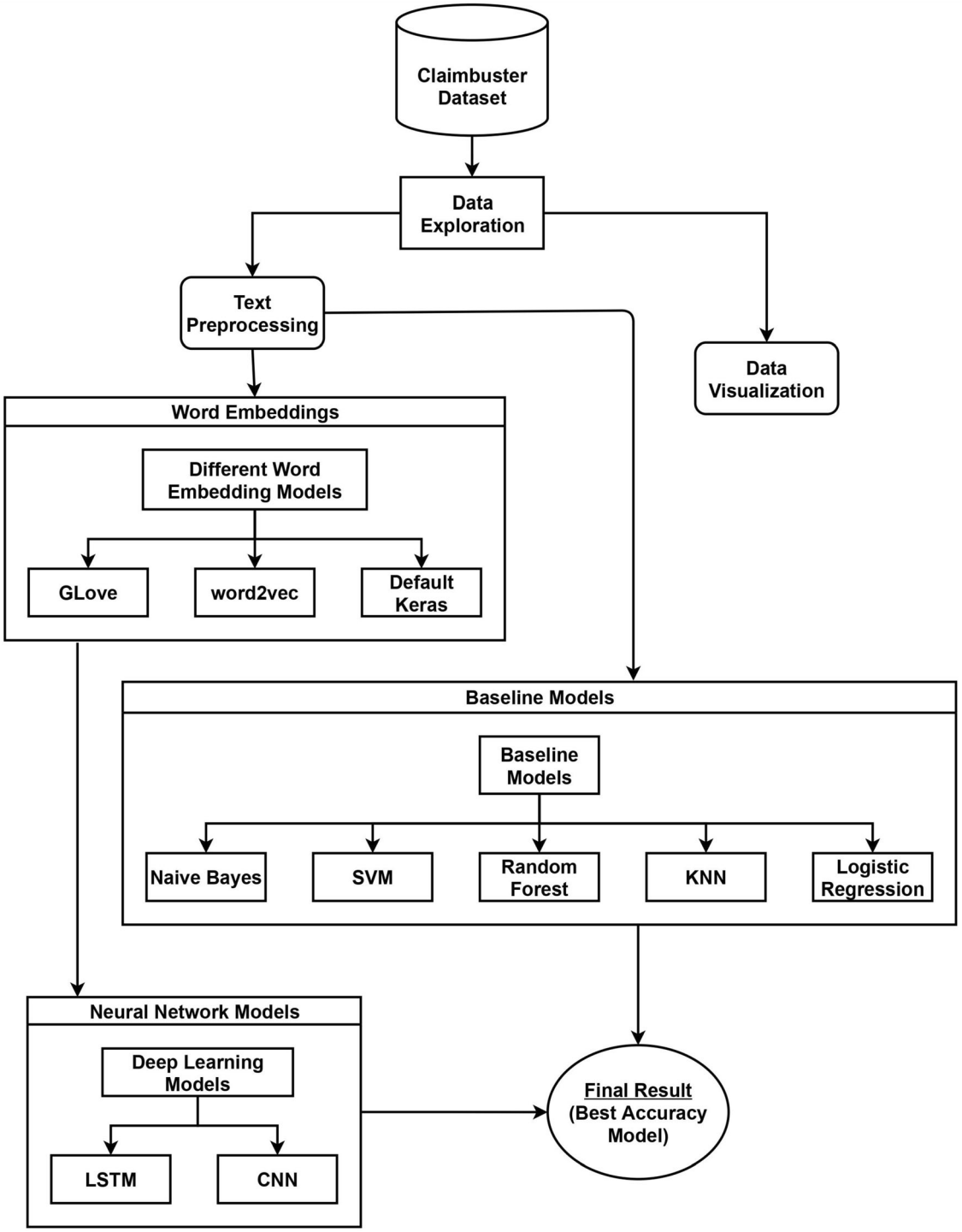

Fig. 12 Workflow of Experiment Design

– GRU: This is our proposed G2CW framework comprising 1 GRU layer with 100 units, followed by 3 units of dense layer and softmax, as explained earlier in the paper.

Table 11 shows the weighted average values of precision, recall, and F1-score of the best-performing models that performed better than the other deep learning models. The weighted average is obtained following the number of records in each class. It turns out that CNN_2 with Word2Vec embedding, and CNN_3, LSTM_3, LSTM_4, and LSTM_5 with glove embedding are the models that

Table 7 Results of our implementations of the baseline models on the ClaimBuster dataset using only sentence extracted features (*W*) and sentence plus POS features (*W_P*)

Model	Features	P_wavg	R_wavg	F1_wavg
SVM	W	0.86	0.86	0.84
	W_P	0.87	0.88	**0.86**
RFC	W	0.56	0.68	0.61
	W_P	0.84	0.85	0.82
NBC	W	0.8	0.83	0.79
	W_P	0.87	0.87	0.84
KNN	W	0.72	0.72	0.61
	W_P	0.75	0.72	0.61
LR	W	0.8	0.84	0.81
	W_P	0.67	0.72	0.67

Table 8 Deep learning model loss and accuracy on the ClaimBuster dataset

Model	Embedding	Train		Validation		Test	
		Loss	Acc	Loss	Acc	Loss	Acc
CNN_1	Default Keras	0.37	0.90	0.90	0.69	0.39	0.90
	Word2vec	0.42	0.88	0.82	0.71	0.35	0.91
	Glove	0.45	0.87	0.79	0.72	0.34	0.91
CNN_4	Default Keras	0.55	0.83	0.80	0.70	0.41	0.89
	Word2vec	0.52	0.84	0.75	0.72	0.36	**0.92**
	Glove	0.45	0.86	0.84	0.70	0.37	0.90
CNN_5	Default Keras	0.54	0.83	0.81	0.70	0.42	0.89
	Word2vec	0.46	0.86	0.81	0.70	0.36	0.91
	Glove	0.46	0.86	0.79	0.72	0.35	0.91
CNN_6	Default Keras	0.61	0.83	0.84	0.70	0.48	0.87
	Word2vec	0.57	0.83	0.80	0.70	0.40	0.90
	Glove	0.56	0.82	0.79	0.71	0.40	0.90
LSTM_1	Default Keras	0.68	0.72	0.76	0.67	0.46	0.85
	Word2Vec	0.65	0.75	0.70	0.73	0.40	0.89
	Glove	0.62	0.77	0.68	0.73	0.36	0.90
LSTM_2	Default Keras	0.64	0.74	0.77	0.67	0.45	0.67
	Word2Vec	0.51	0.80	0.66	0.73	0.29	0.90
	Glove	0.46	0.82	0.69	0.73	0.28	**0.91**

perform the best with 92% precision, 92% recall, and 92% F1-score.

All the models, namely CNN_2, CNN_3, LSTM_3, LSTM_4, LSTM_5, and GRU, performed equally well. Therefore, in order to find out the best among these models, we compare them in terms of the amount of training time consumed. Table 12 shows the training time used by all the best-performing models. Moreover, it turns out that among the best-performing models, the GRU-based proposed G2CW framework takes only 12.5 seconds to train, which is the least training time among all the models.

5.3 Results for Indian dataset

This subsection discusses the results obtained by training the models using the ClaimBuster dataset and testing the IndianClaims dataset.

5.3.1 Baseline models

We summarize our results in Table 13 by presenting average precision, recall, and F1-score values for the IndianClaims dataset. It turns out that the naive Bayes classifier (NBC) with features as *W_P* gives the maximum F1-score of 0.68 among all the baseline models explored.

5.3.2 Deep learning models

We experiment with CNN and LSTM-based models and summarize the results too.

Table 14 shows the loss and accuracy during the validation and testing phase obtained from the various deep learning models on the IndianClaims dataset. Among the CNN-based models, we observe that CNN_1 with Word2vec gives the best accuracy of 72%, which has three convolutional layers with 100 filters, three max pool layers, and three units dense layer (output layer) with softmax activation function. Among the LSTM-based models, we find that LSTM_4 with glove performs better with an accuracy of 67%.

Table 15 shows the class-wise precision, recall, and F1-score for the classes, namely NFS, UFS, and CFS, when we run different deep learning models for testing the IndianClaims dataset. For the NFS class, the maximum precision of 94% is given by the LSTM_4 model with glove as the embedding layer. It turns out that CNN_3 gives the maximum recall of 85% when glove embedding is used. CNN_3 model that captures word dependencies gives the best F1-score of 84% when glove embeddings are used. For the UFS class, the maximum precision of 39% is given by the LSTM_4 model with the Default Keras embedding layer. The maximum recall of 37% is obtained using the LSTM_5 model with glove embedding. Moreover, the maximum F1-score of 27% is achieved by using the LSTM_5 model with glove embeddings. For the CFS class, the maximum precision of 85% is obtained using the LSTM_5 model with glove embeddings. LSTM_1 model gives recall of 99% with glove embeddings. Lastly, the LSTM_5 model gives the best F1-score of 91% with glove embeddings.

Table 9 Precision, Recall, and F1-score for each class in the ClaimBuster dataset

Model	Embedding	Precision			Recall			F1-score		
		NFS	UFS	CFS	NFS	UFS	CFS	NFS	UFS	CFS
CNN_1	Default Keras	0.73	0.84	0.92	0.48	0.82	0.96	0.58	0.83	0.94
	Word2vec	0.78	0.87	0.93	**0.56**	0.83	0.96	0.65	0.85	0.95
	Glove	0.79	0.88	0.93	**0.56**	0.84	0.96	0.65	0.86	0.95
CNN_4	Default Keras	0.93	0.84	0.91	0.21	0.82	**0.98**	0.34	0.83	0.94
	Word2vec	0.82	0.89	0.93	0.52	0.84	0.97	0.64	0.86	0.95
	Glove	0.72	**0.90**	0.91	0.46	0.79	**0.98**	0.56	0.84	0.94
CNN_5	Default Keras	0.75	0.82	0.91	0.19	0.82	0.97	0.30	0.82	0.94
	Word2vec	0.74	0.88	0.92	0.54	0.82	0.97	0.62	0.85	0.95
	Glove	0.72	0.89	0.93	0.54	0.85	0.97	0.62	**0.87**	0.95
CNN_6	Default Keras	**1.00**	0.00	0.89	0.10	0.81	0.96	0.17	0.80	0.92
	Word2vec	0.76	0.87	0.92	0.44	0.82	0.97	0.56	0.84	0.94
	Glove	0.86	0.88	0.91	0.38	0.79	**0.98**	0.53	0.83	0.94
LSTM_1	Default Keras	0.45	0.85	0.86	0.16	0.68	0.96	0.24	0.75	0.91
	Word2Vec	0.53	0.81	0.95	0.48	0.89	0.92	0.50	0.85	0.94
	Glove	0.72	0.82	**0.96**	0.60	**0.92**	0.93	**0.66**	**0.87**	0.95
LSTM_2	Default Keras	0.37	0.80	0.87	0.21	0.68	0.95	0.27	0.73	0.91
	Word2Vec	0.52	0.88	0.94	0.54	0.84	0.95	0.53	0.86	0.95
	Glove	0.59	0.88	**0.96**	0.70	0.83	0.96	0.64	0.85	**0.96**

Table 10 Weighted average values of deep learning models on the ClaimBuster dataset

Model	Embedding	p_wavg	r_wavg	f1_wavg
CNN_1	Default Keras	0.89	0.9	0.89
	Word2vec	0.91	**0.91**	**0.91**
	GloVe	0.91	**0.91**	**0.91**
CNN_4	Default Keras	0.89	0.89	0.88
	Word2vec	0.91	**0.91**	**0.91**
	GloVe	0.9	0.9	0.9
CNN_5	Default Keras	0.88	0.89	0.87
	Word2vec	0.9	0.9	0.9
	GloVe	0.91	**0.91**	**0.91**
CNN_6	Default Keras	0.88	0.87	0.85
	Word2vec	0.9	0.9	0.9
	GloVe	0.9	0.9	0.9
LSTM_1	Default Keras	0.83	0.85	0.83
	Word2Vec	0.89	0.89	0.89
	GloVe	0.91	**0.91**	**0.91**
LSTM_2	Default Keras	0.84	0.86	0.84
	Word2Vec	0.9	0.9	0.9
	GloVe	**0.92**	**0.91**	**0.91**

Table 11 Weighted Average Values of best-performing models on ClaimBuster dataset

Model	Embedding	p_wavg	r_wavg	f1_wavg
CNN_2	Default Keras	0.9	0.9	0.89
	Word2vec	0.92	0.92	0.92
	Glove	0.9	0.9	0.9
CNN_3	Default Keras	0.9	0.9	0.9
	Word2vec	0.9	0.91	0.9
	Glove	0.92	0.92	0.92
LSTM_3	Default Keras	0.82	0.83	0.83
	Word2Vec	0.91	0.91	0.91
	Glove	0.92	0.92	0.92
LSTM_4	Default Keras	0.84	0.86	0.84
	Word2Vec	0.91	0.92	0.91
	Glove	0.92	0.92	0.92
LSTM_5	Default Keras	0.83	0.84	0.83
	Word2Vec	0.92	0.91	0.91
	Glove	0.92	0.92	0.92
G2CW	Default Keras	0.8	0.83	0.81
	Word2Vec	0.91	0.91	0.91
	Glove	**0.92**	**0.92**	**0.92**

Table 16 shows the weighted average values of precision, recall, and F1-score from the various deep learning models on the IndianClaims dataset we explored. The LSTM_4 model gives the maximum weighted average precision of 73% with glove embedding. CNN_1 and CNN_4 give the best value of weighted average recall of

Table 12 Time analysis of best-performing models on ClaimBuster dataset

Model	Embedding	p_wavg	r_wavg	f1_wavg	Time(in sec)
CNN_2	Word2vec	0.92	0.92	0.92	27
CNN_3	Glove	0.92	0.92	0.92	26
LSTM_3	Glove	0.92	0.92	0.92	17.8
LSTM_4	Glove	0.92	0.92	0.92	18.4
LSTM_5	Glove	0.92	0.92	0.92	85
G2CW	**Glove**	**0.92**	**0.92**	**0.92**	**12.5**

Table 13 Results of our implementations of the baseline models on the IndianClaims dataset using only sentence extracted features (*W*) and sentence plus POS features (*W_P*)

Model	Features	P_wavg	R_wavg	F1_wavg
SVM	W	0.54	0.63	0.57
	W_P	0.63	0.69	0.64
RFC	W	0.64	0.68	0.64
	W_P	0.65	0.65	0.65
NBC	W	0.73	0.72	0.66
	W_P	0.74	0.73	**0.68**
KNN	W	0.60	0.60	0.57
	W_P	0.60	0.66	0.62
LR	W	0.59	0.60	0.55
	W_P	0.63	0.67	0.62

71% with Word2Vec embedding and CNN_2 with glove embedding. Similarly, we achieve a maximum weighted average F1-score of 69% using CNN_3, CNN_4, and LSTM_4 with Word2Vec embedding, and CNN_1, CNN_2, CNN_3, CNN_5, and LSTM_4 with glove embedding.

5.3.3 Best-performing models

In addition to the deep-learning models discussed in the baseline and deep-learning section, we explain the deep-learning model that gave us the best result for the IndianClaims dataset. Recall that our proposed G2CW framework based on GRU comprises 1 GRU layer with 100 units, followed by 3 units of dense and softmax units.

Table 17 shows the weighted average values of precision, recall, and F1-score of the best-performing models that performed better than the other deep learning models stated in the previous section for testing the Indian Dataset. The weighted average is obtained following the number of records in each class. It turns out that CNN_6 with Word2Vec embedding and CNN_6 and LSTM_2 with glove embedding are the models that perform the best with a 70% F1-score. Our proposed G2CW framework with glove embedding performs with 72% precision, 69% recall, and 70% F1-score. The models, namely CNN_6, LSTM_2, and G2CW framework, performed equally well. Therefore, to find out the best among these models, we compare them in terms of the amount of training time consumed.

Table 18 shows the training time used by all the best-performing models to test the Indian dataset. Moreover, it turns out that among the best-performing models, the GRU-based proposed G2CW framework takes only 8.12 seconds to train, which is the least training time among all the models.

Table 14 Deep learning model loss and accuracy on IndianClaims dataset

Model	Embedding	Train		Validation		Test	
		Loss	Acc	Loss	Acc	Loss	Acc
CNN_1	Default Keras	0.50	0.85	0.82	0.70	0.89	0.68
	Word2vec	0.65	0.82	0.83	0.71	0.88	**0.72**
	Glove	0.47	0.86	0.74	0.73	0.88	0.69
CNN_2	Default Keras	0.57	0.82	0.79	0.70	0.88	0.69
	Word2vec	0.54	0.83	0.74	0.73	0.83	0.70
	Glove	0.58	0.81	0.72	0.73	0.78	0.71
CNN_3	Default Keras	0.49	0.85	0.80	0.71	0.88	0.68
	Word2vec	0.50	0.85	0.75	0.72	0.86	0.70
	Glove	0.47	0.85	0.73	0.72	0.87	0.69
CNN_4	Default Keras	0.56	0.83	0.80	0.70	0.87	0.69
	Word2vec	0.55	0.83	0.74	0.74	0.83	0.71
	Glove	0.56	0.82	0.72	0.73	0.81	0.70
CNN_5	Default Keras	0.55	0.83	0.80	0.71	0.86	0.68
	Word2vec	0.50	0.85	0.75	0.71	0.86	0.69
	Glove	0.5	0.84	0.73	0.73	0.83	0.70
LSTM_1	Default Keras	0.69	0.73	0.77	0.67	0.94	0.61
	Word2vec	0.60	0.78	0.67	0.73	0.89	0.66
	Glove	0.58	0.78	0.68	0.73	0.85	0.65
LSTM_3	Default Keras	0.64	0.74	0.78	0.67	0.99	0.62
	Word2vec	0.49	0.81	0.66	0.73	0.84	0.66
	Glove	0.46	0.82	0.70	0.73	0.81	0.66
LSTM_4	Default Keras	0.66	0.72	0.78	0.66	0.78	0.68
	Word2vec	0.51	0.80	0.67	0.73	0.78	0.66
	Glove	0.25	0.91	0.84	0.70	0.80	**0.67**
LSTM_5	Default Keras	0.62	0.74	0.77	0.66	0.89	0.61
	Word2vec	0.48	0.81	0.67	0.72	0.89	0.57
	Glove	0.47	0.81	0.66	0.73	0.91	0.64

Table 15 Precision, recall, and F1-score for each Class in IndianClaims dataset

Model	Embedding	Precision			Recall			F1-score		
		NFS	UFS	CFS	NFS	UFS	CFS	NFS	UFS	CFS
CNN_1	Default Keras	0.76	0.2	0.6	0.8	0.05	0.68	0.78	0.08	0.64
	Word2vec	0.8	0	0.63	0.81	0	0.79	0.8	0	0.7
	Glove	0.82	0.21	0.63	0.84	0.16	0.65	0.83	0.18	0.64
CNN_2	Default Keras	0.75	0	0.63	0.83	0	0.7	0.79	0	0.66
	Word2vec	0.81	0.14	0.61	0.81	0.04	0.73	0.81	0.06	0.67
	Glove	0.83	0.16	0.63	0.83	0.06	0.73	0.83	0.09	0.68
CNN_3	Default Keras	0.76	0.14	0.61	0.81	0.03	0.69	0.79	0.05	0.65
	Word2vec	0.81	0.26	0.63	0.81	0.13	0.71	0.81	0.17	0.67
	Glove	0.83	0.18	0.63	**0.85**	0.15	0.64	**0.84**	0.16	0.63
CNN_4	Default Keras	0.75	0	0.63	0.83	0	0.69	0.79	0	0.66
	Word2vec	0.83	0.14	0.62	0.8	0.04	0.77	0.81	0.06	0.69
	Glove	0.82	0.15	0.62	0.82	0.06	0.71	0.82	0.09	0.66
CNN_5	Default Keras	0.75	0.12	0.61	0.81	0.01	0.7	0.78	0.02	0.65
	Word2vec	0.79	0.16	0.62	0.83	0.07	0.68	0.81	0.1	0.65
	Glove	0.83	0.19	0.62	0.83	0.12	0.69	0.83	0.15	0.66
LSTM_1	Default Keras	0.75	0	0.53	0.56	0	0.92	0.64	0	0.67
	Word2vec	0.73	0.17	0.76	0.58	0.17	0.96	0.64	0.17	0.85
	Glove	0.74	0.23	0.78	0.5	0.31	**0.99**	0.6	0.26	0.87
LSTM_3	Default Keras	0.72	0.33	0.56	0.6	0.03	0.89	0.66	0.05	0.69
	Word2vec	0.76	0.21	0.76	0.54	0.26	0.97	0.63	0.23	0.86
	Glove	0.73	0.22	0.81	0.54	0.28	0.98	0.62	0.25	0.88
LSTM_4	Default Keras	0.82	**0.39**	0.58	0.71	0.07	0.81	0.76	0.12	0.68
	Word2vec	0.92	0.13	0.62	0.77	0.2	0.66	**0.84**	0.15	0.64
	Glove	**0.94**	0.17	0.6	0.76	0.29	0.64	**0.84**	0.21	0.62
LSTM_5	Default Keras	0.68	0	0.57	0.65	0	0.82	0.66	0	0.67
	Word2vec	0.73	0.19	0.65	0.37	0.25	0.98	0.49	0.21	0.78
	Glove	0.73	0.21	**0.85**	0.47	**0.37**	0.98	0.57	**0.27**	**0.91**

6 Conclusion and future scope

This work proposed a glove embedding-based GRU architecture, referred to as the G2CW framework, for detecting check-worthy sentences. Our proposed approach gives an F1-score of 0.92, which outperforms the baseline F1-score of 0.81 from Hassan et al. [13]. Our experiments found that other deep learning architectures performed equally well as the G2CW framework. However, the G2CW framework took the least amount of training time. We evaluated the G2CW framework on a standard ClaimBuster dataset curated by Hassan et al. [13], and also performed transfer learning experiments on a self-curated IndianClaims dataset. Our work will be helpful for researchers in the field of fact-checking. G2CW framework can be used to detect whether a sentence is worthy of fact-

Table 16 Weighted average values of deep learning models on IndianClaims dataset

Model	Embedding	p_wavg	r_wavg	f1_wavg
CNN_1	Default Keras	0.64	0.68	0.65
	Word2vec	0.65	**0.71**	0.68
	Glove	0.68	0.69	**0.69**
CNN_2	Default Keras	0.62	0.69	0.66
	Word2vec	0.66	0.7	0.68
	Glove	0.68	**0.71**	**0.69**
CNN_3	Default Keras	0.64	0.68	0.66
	Word2vec	0.68	0.7	**0.69**
	Glove	0.68	0.69	**0.69**
CNN_4	Default Keras	0.62	0.69	0.65
	Word2vec	0.68	**0.71**	**0.69**
	Glove	0.67	0.7	0.68
CNN_5	Default Keras	0.63	0.69	0.65
	Word2vec	0.66	0.69	0.67
	Glove	0.68	0.7	**0.69**
LSTM_1	Default Keras	0.55	0.61	0.56
	Word2vec	0.65	0.66	0.65
	Glove	0.68	0.65	0.65
LSTM_3	Default Keras	0.6	0.62	0.58
	Word2vec	0.68	0.66	0.65
	Glove	0.68	0.67	0.66
LSTM_4	Default Keras	0.68	0.68	0.66
	Word2vec	0.72	0.67	**0.69**
	Glove	**0.73**	0.66	**0.69**
LSTM_5	Default Keras	0.53	0.61	0.57
	Word2vec	0.62	0.58	0.55
	Glove	0.7	0.64	0.65

Table 17 Weighted average values of best-performing model on the IndianClaims dataset

Model	Embedding	p_wavg	r_wavg	f1_wavg
CNN_6	Default Keras	0.66	0.7	0.66
	Word2vec	0.7	**0.73**	**0.7**
	Glove	0.7	**0.73**	**0.7**
LSTM_2	Default Keras	0.57	0.6	0.58
	Word2vec	0.67	0.64	0.65
	Glove	**0.74**	0.68	**0.7**
G2CW	Default Keras	0.57	0.64	0.59
	Word2vec	0.67	0.7	0.68
	Glove	0.72	0.69	**0.7**

Table 18 Time analysis of best-performing model on the IndianClaims dataset

Model	Embedding	p_wavg	r_wavg	f1_wavg	Time(in sec)
CNN6	Word2vec	0.7	0.73	0.7	836
CNN6	Glove	0.7	0.73	0.7	681
LSTM2	Glove	0.74	0.68	0.7	13.3
G2CW	**Glove**	**0.72**	**0.69**	**0.7**	**8.12**

checking. In this manner, only fact-worthy sentences can be further passed to the fact-checking mechanisms.

Data availability The data related to this work shall be made available on reasonable request.

Declarations

Conflict of interest There are no conflicts of interest.

References

1. Arslan F, Hassan N, Li C, Tremayne M (2020) A benchmark dataset of check-worthy factual claims. Proc Int AAAI Conf Web Soc Media 14:821–829
2. Atanasova P, Nakov P, Màrquez L, Barrón-Cedeño A, Karadzhov G, Mihaylova T, Mohtarami M, Glass J (2019) Automatic fact-checking using context and discourse information. J Data Inf Qual (JDIQ) 11(3):1–27
3. Atanasova P, Simonsen JG, Lioma C, Augenstein I (2020) Generating fact checking explanations. arXiv preprint arXiv:2004.05773
4. Elsayed T, Nakov P, Barrón-Cedeno A, Hasanain M, Suwaileh R, Da San Martino G, Atanasova P (2019) Checkthat! at clef 2019: Automatic identification and verification of claims. In: European conference on information retrieval, Springer, pp 309–315
5. Gantz J (2007) Reinsel D (2012) The digital universe in 2020: big data, bigger digital shadows, and biggest growth in the far east. IDC iView: IDC Analyze the future 2012:1–16
6. Graves L (2013) Deciding what's true: fact-checking journalism and the new ecology of news. Columbia University
7. Graves L, Amazeen MA (2019) Fact-checking as idea and practice in journalism. In: oxford Research Encyclopedia of Communication
8. Hansen C, Hansen C, Alstrup S, Grue Simonsen J, Lioma C (2019) Neural check-worthiness ranking with weak supervision: finding sentences for fact-checking. In: Companion proceedings of the 2019 world wide web conference, pp 994–1000
9. Haque MM, Yousuf M, Arman Z, Rony MMU, Alam AS, Hasan KM, Islam MK, Hassan N (2018) Fact-checking initiatives in bangladesh, india, and nepal: a study of user engagement and challenges. arXiv preprint arXiv:1811.01806
10. Hassan N, Adair B, Hamilton JT, Li C, Tremayne M, Yang J, Yu C (2015) The quest to automate fact-checking. In: Proceedings of the 2015 computation+ journalism symposium
11. Hassan N, Tremayne M, Arslan F, Li C (2016) Comparing automated factual claim detection against judgments of

journalism organizations. In: Computation+ Journalism Symposium, pp 1–5
12. Hassan N, Arslan F, Li C, Tremayne M (2017a) Toward automated fact-checking: Detecting check-worthy factual claims by claimbuster. In: Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining, pp 1803–1812
13. Hassan N, Zhang G, Arslan F, Caraballo J, Jimenez D, Gawsane S, Hasan S, Joseph M, Kulkarni A, Nayak AK et al (2017) Claimbuster: the first-ever end-to-end fact-checking system. Proc VLDB Endowment 10(12):1945–1948
14. Jaradat I, Gencheva P, Barrón-Cedeño A, Màrquez L, Nakov P (2018) Claimrank: detecting check-worthy claims in arabic and english. arXiv preprint arXiv:1804.07587
15. Khan N, Yaqoob I, Hashem IAT, Inayat Z, Mahmoud Ali WK, Alam M, Shiraz M, Gani A (2014) Big data: survey, technologies, opportunities, and challenges. The scientific world journal 2014
16. Konstantinovskiy L, Price O, Babakar M, Zubiaga A (2018) Towards automated factchecking: developing an annotation schema and benchmark for consistent automated claim detection. arXiv preprint arXiv:1809.08193
17. Konstantinovskiy L, Price O, Babakar M, Zubiaga A (2021) Toward automated factchecking: developing an annotation schema and benchmark for consistent automated claim detection. Dig Threats: Res Pract 2(2):1–16
18. Leeper TJ, Slothuus R (2014) Political parties, motivated reasoning, and public opinion formation. Polit Psychol 35:129–156
19. Melo P, Messias J, Resende G, Garimella K, Almeida J, Benevenuto F (2019) Whatsapp monitor: a fact-checking system for whatsapp. Proc Int AAAI Conf Web and Soc Media 13:676–677
20. Mihaylova T, Nakov P, Màrquez L, Barrón-Cedeño A, Mohtarami M, Karadzhov G, Glass J (2018) Fact checking in community forums. In: Proceedings of the AAAI conference on artificial intelligence, vol 32
21. Miranda S, Nogueira D, Mendes A, Vlachos A, Secker A, Garrett R, Mitchel J, Marinho Z (2019) Automated fact checking in the news room. In: The World Wide Web Conference, pp 3579–3583
22. Nadeem M, Fang W, Xu B, Mohtarami M, Glass J (2019) Fakta: an automatic end-to-end fact checking system. arXiv preprint arXiv:1906.04164
23. Nakov P, Barrón-Cedeno A, Elsayed T, Suwaileh R, Màrquez L, Zaghouani W, Atanasova P, Kyuchukov S, Da San Martino G (2018) Overview of the clef-2018 checkthat! lab on automatic identification and verification of political claims. In: International conference of the cross-language evaluation forum for european languages, Springer, pp 372–387
24. Patwa P, Sharma S, PYKL S, Guptha V, Kumari G, Akhtar MS, Ekbal A, Das A, Chakraborty T (2020) Fighting an infodemic: Covid-19 fake news dataset. arXiv preprint arXiv:2011.03327
25. Thorne J, Vlachos A (2018) Automated fact checking: task formulations, methods and future directions. arXiv preprint arXiv:1806.07687
26. Vo N, Lee K (2018) The rise of guardians: Fact-checking url recommendation to combat fake news. In: The 41st international ACM SIGIR conference on research & development in information retrieval, pp 275–284
27. Watts DJ, Dodds PS (2007) Influentials, networks, and public opinion formation. J Consum Res 34(4):441–458
28. Zuo C, Karakas A, Banerjee R (2018) A hybrid recognition system for check-worthy claims using heuristics and supervised learning. In: CLEF (Working Notes)