COMMENTATOR ✍️: A Code-mixed Multilingual Text Annotation Framework

Rajvee Sheth[†], **Shubh Nisar**[⋆], **Heenaben Prajapati**[†],
Himanshu Beniwal[†], **Mayank Singh**[†],

[†]Indian Institute of Technology Gandhinagar, [⋆]North Carolina State University
Correspondence: lingo@iitgn.ac.in

Abstract

As the NLP community increasingly addresses challenges associated with multilingualism, robust annotation tools are essential to handle multilingual datasets efficiently. In this paper, we introduce a **co**de-**m**ixed **m**ultilingual t**e**xt a**n**no**tat**ion framew**or**k, COMMENTATOR, specifically designed for annotating code-mixed text. The tool demonstrates its effectiveness in token-level and sentence-level language annotation tasks for **Hinglish** text. We perform robust qualitative human-based evaluations to showcase COMMENTATOR led to **5x** faster annotations than the best baseline. Our code is publicly available at `https://github.com/lingo-iitgn/commentator`. The demonstration video is available at `https://bit.ly/commentator_video`.

1 Introduction

Code mixing is prevalent in informal conversations and in social media, where elements from different languages are interwoven within a single sentence. A representative example in Hinglish such as "*I am feeling very thand today, so I'll wear a sweater.*" (In this sentence, "*thand*" is a Hindi word meaning "*cold*", while the rest of the sentence is in English), demonstrating seamless integration of Hindi and English. A major challenge in NLP research is the scarcity of high-quality datasets, which require extensive manual efforts, significant time, domain expertise, and linguistic understanding, as highlighted by Hovy and Lavid (2010). The rise of social media has further complicated annotation tasks due to non-standard grammar, platform-specific tokens, and neologisms (Shahi and Majchrzak, 2022). Annotating these datasets presents unique challenges, including ensuring data consistency, efficiently managing large datasets, mitigating annotator biases, and reporting poor-quality instances. Existing annotation tools often fail to address these diverse issues effectively.

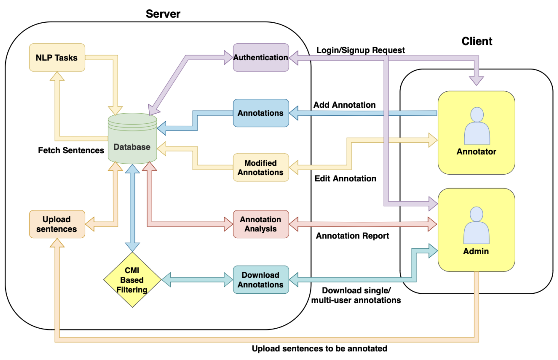


Figure 1: COMMENTATOR Framework.

This paper introduces COMMENTATOR, a robust annotation framework designed for multiple code-mixed annotation tasks. The current version[1] of COMMENTATOR supports two token-level annotation tasks, **Language Identification**, **POS tagging**, and sentence-level **Matrix Language Identification**. While COMMENTATOR has already been used to generate a large number of annotations (more than 100K) in our ongoing project[2], these are not part of the current demo paper. The focus of this paper is to present the capabilities and initial functionalities of the framework. Figure 1 presents the framework COMMENTATOR.

We evaluate COMMENTATOR by comparing its features and performance against five state-of-the-art text annotation tools, (i) YEDDA (Yang et al., 2018), (ii) MarkUp (Dobbie et al., 2021), (iii) INCEpTION (Klie et al., 2018), (iv) UBIAI[3] and (v) GATE (Cunningham et al., 1996). The major perceived capabilities (see Section 4.1) of COMMENTATOR are (i) simplicity in navigation and performing basic actions, (ii) task-specific recommendations to improve user productivity and ease the

[1]As a continual development effort, it will be further extended to three more popular code-mixing tasks NER, Spell Correction and Normalization, and Machine Translation.
[2]URL available on our Github.
[3]`https://ubiai.tools/`

annotation process, (iii) quick cloud or local setup with minimal dependency requirements, (iv) promoting iterative refinement and quality control by integrating annotator feedback, (v) simple admin interface for uploading data, monitoring progress and post-annotation data analysis, and (vi) parallel annotations enabling multiple users to work on the same project simultaneously. Furthermore, Section 4.2 demonstrates an annotation speed increase of nearly 5x compared to the nearest SOTA baseline. This speed gain can be further enhanced by incorporating more advanced code-mixed libraries.

In addition, the codebase, the demo website with a detailed installation guide, and some Hinglish sample instances are available on GitHub[4]. Currently, the functionality is tailored for Hinglish, but it can be extended to support any language pair.

2 Existing Text Annotation Frameworks

Text annotation tools are vital in NLP for creating annotated datasets for training and evaluating machine learning models. This summary reviews several key tools, each with unique features and limitations.

2.1 Web-based Annotation Tools

These tools have been created to provide annotation environments independent of operating systems. Some of the web-based annotation tools are: ***(1)*** *MarkUp* improves annotation speed and accuracy using NLP and active learning but requires re-annotation for updates and has unreliable collaboration features (Dobbie et al., 2021), ***(2)*** *INCEpTION* offers a versatile platform for semantic and interactive annotation but struggles with session timeouts and updating annotations (Klie et al., 2018), and lastly, ***(3)*** *UBIAI* provides advanced cloud-based NLP functions but faces problems with incorrect entity assignments and model integration (ubi, 2022).

2.2 Locally-hosted Tools

These tools can be installed on a local machine and offer more robust features or better performance for large datasets. Some of the locally hosted tools are: ***(1)*** *YEDDA* is an open source tool that enhances annotation efficiency and supports collaborative and administrative functions, though it has limitations in customization and can break tokens during annotation (Yang et al., 2018), ***(2)*** *GATE* is an open-source tool known for its real-time collaboration, but it is complicated to configure and slow with API requests (Bontcheva et al., 2013), ***(3)*** *BRAT* is user-friendly for entity recognition and relationship annotation but lacks active learning, automatic suggestions, and does not provide post-annotation analysis features. Additionally it lacks a dedicated admin interface for user management and annotation monitoring, limiting its overall effectiveness. (Stenetorp et al., 2012), ***(4)*** *Prodigy* integrates with machine learning workflows and supports active learning but requires a commercial license (Montani and Honnibal, 2018), and ***(5)*** *Doccano* is an open-source tool with a customizable interface for various annotation tasks but lacks advanced features like real-time collaboration (Nakayama et al., 2018). Additional tools include ***(6)*** *Knowtator*, designed for biomedical annotations within *Protégé*, but requires significant manual setup (Ogren, 2006), ***(7)*** *WordFreak*, which is flexible but challenging for non-technical users (Morton and LaCivita, 2003), ***(8)*** *Anafora*, known for its efficiency in biomedical annotation but lacking integration with machine learning models (Chen and Styler, 2013), ***(9)*** *Atomic*, which is modular and powerful but requires extensive customization (Druskat et al., 2014), lastly, ***(10)*** *WebAnno* supports a wide range of annotation tasks and collaborative work, but encounters performance issues with large datasets (Yimam et al., 2013).

While these tools offer diverse functionalities, each exhibits limitations that affect efficiency and usability. Most state-of-the-art frameworks are either paid or closed-source and do not support annotator feedback. Additionally, the majority do not enable parallel annotations over the internet and perform poorly when multiple scripts or words from different languages appear in the same sentence. The introduction of COMMENTATOR seeks to address these challenges by providing a robust framework specifically designed for multiple code-mixed annotation tasks.

3 COMMENTATOR

3.1 The Functionalities

The proposed system caters to two types of users: ***(i)*** the annotators and ***(ii)*** the admins. Annotators perform annotation tasks. The admins design the annotation task, employ annotators, administer the annotation task, and process the annotations. Given these roles, we describe the COMMENTATOR functionalities by introducing:

[4]https://github.com/lingo-iitgn/commentator

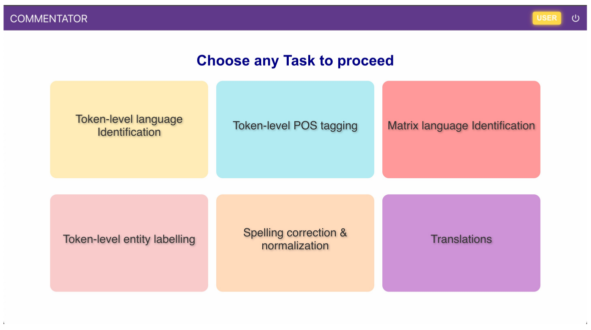

COMMENTATOR
USER
Choose any Task to proceed
Token-level language Identification
Token-level POS tagging
Matrix language Identification
Token-level entity labelling
Spelling correction & normalization
Translations

Figure 2: The Task interface of the COMMENTATOR.

3.1.1 The Annotator Panel

The annotator panel contains three pages:

1. *Landing page*: Figure 2 presents an annotator landing page. Here, the annotators are presented with a selection of several NLP tasks, displayed as clickable options. Selecting a task directs them to the dedicated annotation page for that specific task.
2. *Annotation pages*: We, next, describe annotation pages for the first three tasks:
   - **Token-Level Language Identification (LID):** This task involves identifying the language of individual words (tokens) within a sentence (Figure 3a, point **1**). Each token is pre-assigned a language tag using a state-of-the-art language identification API [5](more details are presented in Section 3.2.2). Annotators can update these tags by clicking the tag button until the desired tag appears. Textual feedback can be entered in the "*Enter Your Feedback Here*" section (Figure 3a, point **3**). Textual feedback is essential to highlight issues with the current sentence. Some issues include grammatically incorrect sentences, incomplete sentences, sensitive/private information, toxic content, etc.
   - **Token-Level Parts-Of-Speech Tagging (POS):** Similar to LID, this task involves identifying the POS tags of individual tokens within a text. Each token is pre-assigned a language tag using a state-of-the-art POS tagging CodeSwitch NLP library [6](more details are presented in Section 3.2.2). In case of incorrect assignment of the tag, the annotators can select the correct tag from a drop-down menu (Figure 4a, point **1**). We do not keep the toggling button feature due to many POS tags. Similarly to LID, annotators can provide feedback (Figure 4a, point **3**).
   - **Matrix Language Identification (MLI):** As shown in Figure 5, this task involves identifying the language that provides the syntactic structure of a code-mixed sentence. Annotators select the matrix language from the multiple supported languages for each sentence (Figure 5, point **1**).

   The primary instructions are present on the left side of the page for each task (See point **2** in Figures 3a, 4a and 5a). Similarly, annotations can be corrected by clicking the "Edit Annotations" button (see point **4** in Figures 3a, 4a and 5a), which redirects to the corresponsing *history and edit* pages (see Figures 3b, 4b and 5b).
3. *History and Edit pages*: Figures 3b, 4b and 5b show a list of previously annotated sentences with timestamps for LID, POS and MLI, respectively. Clicking on a sentence opens the respective annotation page with the previously chosen tags for editing.

3.1.2 The Admin Panel

Figure 6 shows the admin panel. The admin panel performs three major tasks:

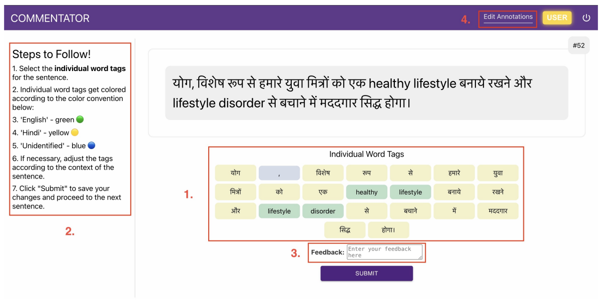
1. *Data upload*: The administrator can upload the source sentences using a CSV file (Figure 6, point **1**).
2. *Annotation analysis*: The administrator can: ***(i)*** analyze the quality of annotations using Cohen's Kappa score for inter-annotator agreement (IAA) (Figure 6, point **3**) and ***(ii)*** analyze the degree of code-mixing in the annotated text using the code-mixing index (CMI) (Das and Gambäck, 2014a)[7](Figure 6, point **2**).
3. *Data download*: The admin can *download* annotations of single/multiple annotators in a CSV file. Admins can select specific tasks from a dropdown menu to customize the data extraction (Figure 6, point **2**) The data download functionality also supports the conditional filtering of data based on *IAA* and *CMI*.

3.2 The Architecture

Figure 1 showcases the highly modular architecture for COMMENTATOR. We describe it using two main

[5]https://github.com/microsoft/LID-tool
[6]https://github.com/sagorbrur/codeswitch

[7]The CMI score ranges from 0 (monolingual) to 100 (highly code-mixed).



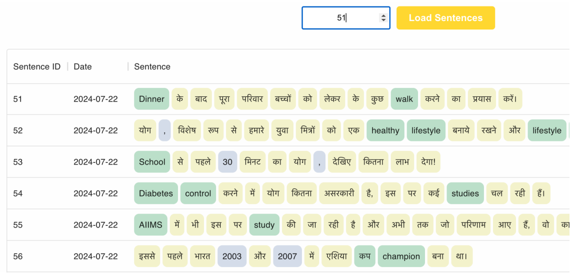
Steps to Follow!
1. Select the **individual word tags** for the sentence.
2. Individual word tags get colored according to the color convention below:
3. 'English' - green 🟢
4. 'Hindi' - yellow 🟡
5. 'Unidentified' - blue 🔵
6. If necessary, adjust the tags according to the context of the sentence.
7. Click "Submit" to save your changes and proceed to the next sentence.

योग, विशेष रूप से हमारे युवा मित्रों को एक healthy lifestyle बनाये रखने और lifestyle disorder से बचाने में मददगार सिद्ध होगा।

Individual Word Tags

योग , विशेष रूप से हमारे युवा मित्रों को एक healthy lifestyle बनाये रखने और lifestyle disorder से बचाने में मददगार सिद्ध होगा।

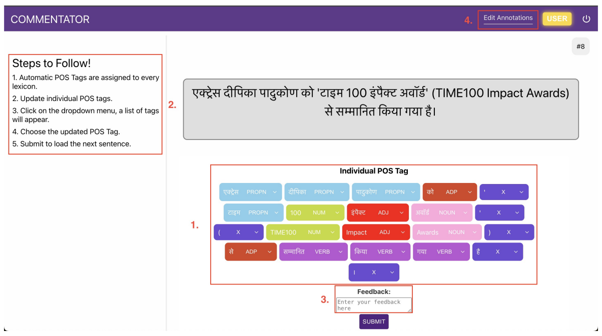
Feedback: Enter your feedback here



Sentence ID	Date	Sentence
51	2024-07-22	Dinner के बाद पूरा परिवार बच्चों को लेकर के कुछ walk करने का प्रयास करें।
52	2024-07-22	योग , विशेष रूप से हमारे युवा मित्रों को एक healthy lifestyle बनाये रखने और lifestyle
53	2024-07-22	School से पहले 30 मिनट का योग , देखिए कितना लाभ देगा!
54	2024-07-22	Diabetes control करने में योग कितना असरकारी है, इस पर कई studies चल रही हैं।
55	2024-07-22	AIIMS में भी इस पर study की जा रही है और अभी तक जो परिणाम आए हैं, वो का
56	2024-07-22	इससे पहले भारत 2003 और 2007 में एशिया कप champion बना था।

(a) (b)

Figure 3: Token-Level Language Identification (LID): (a) annotation page and (b) history and edit page.



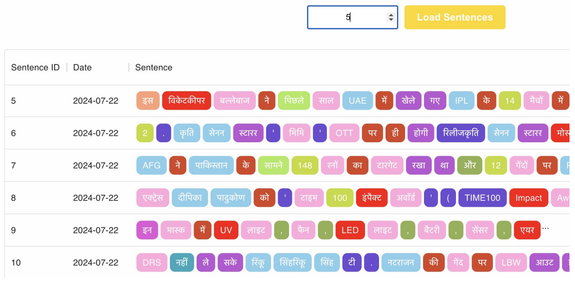
Steps to Follow!
1. Automatic POS Tags are assigned to every lexicon.
2. Update individual POS tags.
3. Click on the dropdown menu, a list of tags will appear.
4. Choose the updated POS Tag.
5. Submit to load the next sentence.

एक्ट्रेस दीपिका पादुकोण को 'टाइम 100 इंपैक्ट अवॉर्ड' (TIME100 Impact Awards) से सम्मानित किया गया है।

Individual POS Tag

एक्ट्रेस PROPN, दीपिका PROPN, पादुकोण PROPN, को ADP, ' X, टाइम PROPN, 100 NUM, इंपैक्ट ADJ, अवॉर्ड NOUN, ' X, (X, TIME100 NUM, Impact ADJ, Awards NOUN,) X, से ADP, सम्मानित VERB, किया VERB, गया VERB, है X, । X

Feedback: Enter your feedback here



Sentence ID	Date	Sentence
5	2024-07-22	इस विकेटकीपर बल्लेबाज ने पिछले साल UAE में खेले गए IPL के 14 मैचों में
6	2024-07-22	2 . कृति सेनन स्टारर ' मिमि ' OTT पर ही होगी रिलीजकृति सेनन स्टारर
7	2024-07-22	AFG ने पाकिस्तान के सामने 148 रनों का टारगेट रखा था और 12 गेंदों पर
8	2024-07-22	एक्ट्रेस दीपिका पादुकोण को ' टाइम 100 इंपैक्ट अवॉर्ड ' (TIME100 Impact
9	2024-07-22	इन मास्क में UV लाइट , फैन , LED लाइट , बैटरी , सेंसर , एयर ...
10	2024-07-22	DRS नहीं ले सके रिंकू सिंहरिंकू सिंह टी . नटराजन की गेंद पर LBW आउट

(a) (b)

Figure 4: Token-Level Parts-Of-Speech Tagging (POS): (a) annotation page and (b) history and edit page.

modules:

3.2.1 Client Module

The client is developed using *ReactJS*[8]. The client module comprises pages for the following functionalities: ***(i)*** User Login, ***(ii)*** User Signup, ***(iii)*** Annotation Panel, and ***(iv)*** History, and ***(v)*** Admin Panel. The user login page is used to log into the portal. The user signup page creates a new annotator account on the portal. The annotation panel is the main landing page that initiates the annotation process for all tasks. The history page lists the annotated sentences by the logged-in annotator for individual tasks.

3.2.2 Server Module

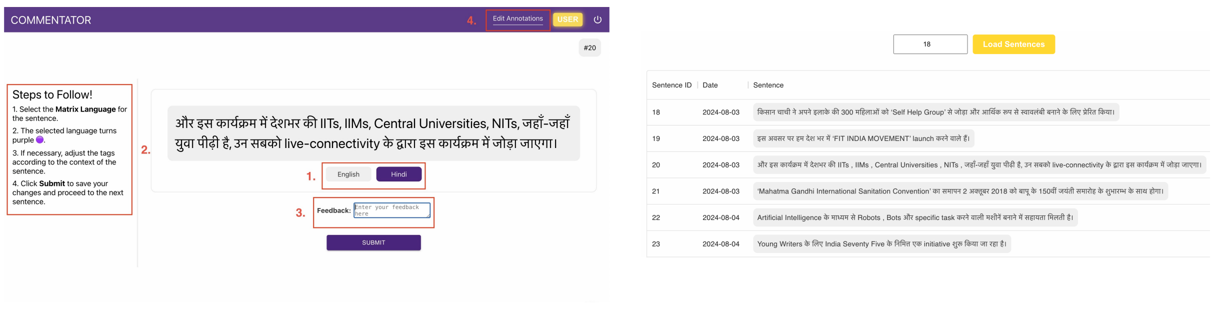
The client is served using a Flask[9] Server. The server performs two major functions: ***(i)*** connection with the database and ***(ii)*** calling task-specific API/libraries. It connects to the *MongoDB* database through a Pymongo library. The MongoDB database can be locally hosted or on the cloud. We use the MongoDB Atlas database[10] hosted locally. In the current setup, we use Microsoft API for LID[11]. For POS, we use the CodeSwitch NLP library. This also demonstrates the flexibility of COMMENTATOR to make web-based API calls or local-hosted library calls based on the task requirements.

4 Experiments

In this section, we perform two human studies to evaluate *COMMENTATOR* against recent state-of-the-art tools to ensure a comprehensive comparison with modern advancements and cutting-edge functionalities: (i) YEDDA (Yang et al., 2018), (ii) MarkUp (Dobbie et al., 2021), (iii) INCEpTION (Klie et al., 2018), (iv) UBIAI[12], and (v) GATE (Bontcheva et al., 2013) (vi) BRAT (Stenetorp et al., 2012). The first study assesses the total time and perceived capabilities during the initial low-level setup and at higher-level annotation tasks (see Section 4.1 for more details). The second study examines the annotation time (see Section 4.2 for more details).

[8]https://reactjs.org
[9]https://flask.palletsprojects.com/en/2.1.x/
[10]https://www.mongodb.com/atlas/database

[11]Existing open source libraries such as Spacy-LangDetect (https://pypi.org/project/spacy-langdetect/) and LangDetect (https://pypi.org/project/langdetect/) showed poor performance
[12]https://ubiai.tools/



COMMENTATOR

4. Edit Annotations USER

#20

Steps to Follow!

1. Select the **Matrix Language** for the sentence.

2. The selected language turns purple 🟣.

3. If necessary, adjust the tags according to the context of the sentence.

4. Click **Submit** to save your changes and proceed to the next sentence.

2. और इस कार्यक्रम में देशभर की IITs, IIMs, Central Universities, NITs, जहाँ-जहाँ युवा पीढ़ी है, उन सबको live-connectivity के द्वारा इस कार्यक्रम में जोड़ा जाएगा।

1. English Hindi

3. Feedback: Enter your feedback here

SUBMIT

18 Load Sentences

Sentence ID	Date	Sentence
18	2024-08-03	किसान चाची ने अपने इलाके की 300 महिलाओं को 'Self Help Group' से जोड़ा और आर्थिक रूप से स्वावलंबी बनाने के लिए प्रेरित किया।
19	2024-08-03	इस अवसर पर हम देश भर में 'FIT INDIA MOVEMENT' launch करने वाले हैं।
20	2024-08-03	और इस कार्यक्रम में देशभर की IITs , IIMs , Central Universities , NITs , जहाँ-जहाँ युवा पीढ़ी है, उन सबको live-connectivity के द्वारा इस कार्यक्रम में जोड़ा जाएगा।
21	2024-08-03	'Mahatma Gandhi International Sanitation Convention' का समापन 2 अक्तूबर 2018 को बापू के 150वीं जयंती समारोह के शुभारम्भ के साथ होगा।
22	2024-08-04	Artificial Intelligence के माध्यम से Robots , Bots और specific task करने वाली मशीनें बनाने में सहायता मिलती है।
23	2024-08-04	Young Writers के लिए India Seventy Five के निमित्त एक initiative शुरू किया जा रहा है।

(a) (b)

Figure 5: Matrix Language Identification (MID): (a) annotation page and (b) history and edit page.

Capabilities	*YEDDA*			*MarkUp*			*INCEpTION*			*UBIAI*			*GATE*			*BRAT*			COMMENTATOR		
	1	2	3	1	2	3	1	2	3	1	2	3	1	2	3	1	2	3	1	2	3
Operational ease	✗	✗	✓	✓	✓	✗	✓	✗	✗	✗	✓	✓	✗	✗	✗	✓	✓	✗	✓	✓	✓
Less dependency requirements	✓	✓	✓	✓	✓	✓	✗	✗	✓	✗	✓	✓	✗	✓	✓	✓	✓	✗	✓	✓	✓
Low latency in API requests	✗	✗	✗	✗	✓	✗	✗	✗	✓	✓	✗	✗	✓	✗	✓	✗	✗	✗	✓	✓	✓
Admin Interface	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✗	✗	✗	✓	✓	✓
System recommendation	✓	✓	✗	✗	✗	✗	✓	✓	✗	✓	✓	✓	✓	✗	✗	✗	✗	✗	✓	✓	✓
Multiple user collaboration	✗	✗	✗	✗	✓	✗	✓	✓	✓	✓	✓	✓	✗	✗	✗	✓	✓	✓	✓	✓	✓
Annotation Refinement and Feedback	✓	✗	✗	✗	✓	✓	✓	✗	✗	✓	✓	✓	✓	✗	✓	✓	✓	✓	✓	✓	✓
Post-annotation analysis	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✗	✗	✗	✓	✓	✓

Table 1: Perceived capabilities by annotators. All annotators perceive all the eight capabilities in COMMENTATOR.

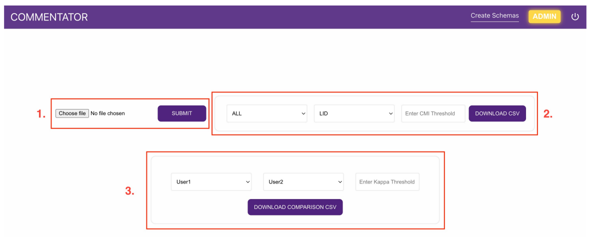


Figure 6: The admin interface of the COMMENTATOR.

4.1 Initial Setup and Perceived Capabilities

We employ three human annotators proficient in English and Hindi with experience using social media platforms such as X (formally 'Twitter'). Additionally, the annotators are graduate students with good programming skills and knowledge of version control systems. Each annotator has a detailed instruction document[13] containing links to execute codebases or access the web user interface, descriptions of tool configurations, annotation processes, and guidelines for recording time.

Each annotator measures the time taken for the initial setup, including installation and configuration. The initial setup includes installation (downloading source code, decompressing, and installing dependencies) and configuration (adding configuration files, sentence loading, and user account creation/login).:

1. *Operational Ease*: A tool demonstrates operational ease when it requires minimal effort for installation, data input, and output. A user-friendly interface with features like color gradients for tag differentiation enhances the annotation experience, leading to more engaging and prolonged usage compared to tools with less visually appealing interfaces.
2. *Less Dependency Requirements*: Annotation tools often require resolving multiple dependencies during installation, which is challenging due to rapid advancements in web frameworks, data processing pipelines, and programming languages. This complexity limits usage, particularly among non-CS users.
3. *Low Latency in API Requests*: Latency is measured as the time to serve the request made by a client. This is the main bottleneck in web-based annotation tools that deal with APIs to serve and process data.
4. *Admin Interface*: The tool should feature an intuitive admin interface for efficient user management, role assignment, and annotation progress monitoring, offering comprehensive control without requiring extensive technical knowledge.
5. *System Recommendation*: Effective system recommendations that use advanced NLP tools and APIs can streamline the annotation process and

[13]https://github.com/lingo-iitgn/commentator/tree/main/Documents

Tools	*Installation*	*Configuration*
YEDDA	**7.66 ± 8.73**	**24.33 ± 32.29**
MarkUp	NA	366.67 ± 47.25
INCEpTION	NA	247.66 ± 39.80
UBIAI	NA	324.33 ± 62.90
GATE	45.67 ± 11.44	125.00 ± 68.07
COMMENTATOR (ours)	173.33 ± 89.93	210.00 ± 81.65

Table 2: Comparison of time taken (mean ± standard deviation) for installation and configuration in seconds. 'NA' corresponds to those web-based tools that cannot be installed on local systems. YEDDA takes the least time to install and configure. COMMENTATOR's configuration time is lower than three popular tools, MarkUp, INCEpTION and UBIAI.

Tools	*LID*	*POS*
YEDDA	757.00 ± 62.27	1370.66 ± 81.24
MarkUp	1192.33 ± 172.77	1579.00 ± 68.86
INCEpTION	1040.66 ± 69.67	1714.66 ± 71.30
UBIAI	690.66 ± 79.43	748.33 ± 91.45
GATE	1118.33 ± 166.20	1579.00 ± 50.61
COMMENTATOR (ours)	**138.33 ± 24.60**	**337.66 ± 25.34**

Table 3: Comparison of time taken (mean ± standard deviation) for annotation in seconds. *POS*, being a highly challenging task than LID, took significantly more time. LID annotations on COMMENTATOR are **5x** faster than the next best tool, UBIAI. Whereas POS annotations on COMMENTATOR are **2x** faster than UBIAI.

reduce the annotation time.

6. *Parallel Annotations*: The tool should support multiple users to work simultaneously on the same dataset, share insights, and maintain consistency across annotations, enhancing overall efficiency and reliability.
7. *Annotation Refinement and Feedback*: The tool must allow annotators to refine and update their annotations easily.
8. *Post-annotation Analysis*: This feature evaluates annotation quality using metrics like inter-annotator agreement, with statistical measures like Cohen's Kappa (it gauges the degree of consistency among annotations), enhancing the reliability and validity of the data. In addition, as the COMMENTATOR largely focuses on the code-mixed domain; integration of metrics like Code-mixing Index (CMI) is highly preferred.

Annotators report each tool's setup time and assign a "Yes/No" label to eight perceived capabilities. Table 2 reports the time taken in seconds for five baselines tool and COMMENTATOR. Overall, YEDDA takes the least time to install and configure. However, Table 1 presents a slightly more distinct picture. COMMENTATOR receives all eight perceived capabilities, while all existing state-of-the-art annotation frameworks, except UIBAI, lack operational ease. Additionally, none of the tools possess a feedback mechanism that allows users to report any inconsistencies during annotations, including identifying noisy or abusive datasets for potential removal. All annotators agree that YEDDA exhibits poor user collaboration capabilities.

4.2 Annotation Time

In the second human study, we recruit three annotators with a good understanding of Hindi and English languages[14]. Each annotator annotates ten Hinglish sentences (available on the project's GitHub page) for token-level language tasks: (i) LID and (ii) POS. Both tasks involve assigning a tag to each token in a sentence. For LID, the tags are *Hindi*, *English*, *Unidentified*. For POS, we follow the list of tags proposed by Singh et al. (2018). This list includes *NOUN*, *PROPN*, *VERB*, *ADJ*, *ADV*, *ADP*, *PRON*, *DET*, *CONJ*, *PART*, *PRON_WH*, *PART_NEG*, *NUM*, and *X*. Here, X denotes foreign words, typos, and abbreviations. Table 3 shows that the libraries that preassign tags enable COMMENTATOR to perform at least five times faster in annotation than the existing tools.

Overall, annotators find that COMMENTATOR takes slightly longer time in initial setup but significantly reduces annotation time and efforts. It showcases good recommendation capability, parallel annotations and post-annotation analysis capabilities.

5 Conclusion and Future Work

We introduce COMMENTATOR, an annotation framework for code-mixed text, and compared it against five-six state-of-the-art annotation tools. COMMENTATOR shows better user collaboration, operational ease, and efficiency, significantly reducing annotation time for tasks like Language Identification and Part-of-Speech tagging. Future plans include expanding COMMENTATOR to support tasks such as sentiment analysis, Q&A, and language generation, making it an even more comprehensive tool for multilingual and code-mixed text annotation.

[14]The three annotators recruited in the first human study are different than these annotators.

6 Ethics

We adhere to the ethical guidelines by ensuring the responsible development and use of our annotation tool. Our project prioritizes annotator well-being, data privacy, and bias mitigation while promoting transparency and inclusivity in NLP research.

References

2022. Ubiai: Nlp annotation tools - automatic text annotation tool.

Kalina Bontcheva, Hamish Cunningham, Ian Roberts, Angus Roberts, Valentin Tablan, Niraj Aswani, and Genevieve Gorrell. 2013. Gate teamware: a web-based, collaborative text annotation framework. *Language Resources and Evaluation*, 47:1007–1029.

Wei-Te Chen and Will Styler. 2013. Anafora: A web-based general purpose annotation tool. In *Proceedings of the 2013 NAACL HLT Demonstration Session*, pages 14–19, Atlanta, Georgia. Association for Computational Linguistics.

Jacob Cohen. 1960. A coefficient of agreement for nominal scales. *Educational and psychological measurement*, 20(1):37–46.

Hamish Cunningham, Yorick Wilks, and Robert Gaizauskas. 1996. Gate-a general architecture for text engineering. In *COLING 1996 Volume 2: The 16th International Conference on Computational Linguistics*.

Amitava Das and Björn Gambäck. 2014a. Identifying languages at the word level in code-mixed Indian social media text. In *Proceedings of the 11th International Conference on Natural Language Processing*, pages 378–387, Goa, India. NLP Association of India.

Amitava Das and Björn Gambäck. 2014b. Identifying languages at the word level in code-mixed indian social media text. In *Proceedings of the 11th International Conference on Natural Language Processing*, pages 378–387.

S Dobbie, H Strafford, WO Pickrell, B Fonferko-Shadrach, C Jones, A Akbari, S Thompson, and A Lacey. 2021. Markup: A web-based annotation tool powered by active learning. *Frontiers in Digital Health*, 3:598916–598916.

Stephan Druskat, Ulrike Gut, Nils Reiter, Stefan Schweter, and Manfred Stede. 2014. Atomic: An open-source tool for working with anaphora in multiple languages. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 71–76.

Kevin Hallgren. 2012. Computing inter-rater reliability for observational data: An overview and tutorial. *Tutorials in Quantitative Methods for Psychology*, 8:23–34.

Eduard Hovy and Julia Lavid. 2010. Towards a 'science'of corpus annotation: a new methodological challenge for corpus linguistics. *International journal of translation*, 22(1):13–36.

Jan-Christoph Klie, Michael Bugert, Beto Boullosa, Richard Eckart de Castilho, and Iryna Gurevych. 2018. The INCEpTION platform: Machine-assisted and knowledge-oriented interactive annotation. In *Proceedings of the 27th International Conference on Computational Linguistics: System Demonstrations*, pages 5–9, Santa Fe, New Mexico. Association for Computational Linguistics.

Ines Montani and Matthew Honnibal. 2018. Prodigy: A new annotation tool for radically efficient machine teaching. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 50–55.

Thomas Morton and Jeremy LaCivita. 2003. WordFreak: An open tool for linguistic annotation. In *Companion Volume of the Proceedings of HLT-NAACL 2003 - Demonstrations*, pages 17–18.

Hiroki Nakayama, Tomoyuki Kubo, Naoki Yoshinaga, and Masaru Kitsuregawa. 2018. Doccano: Text annotation tool for human. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 1–6.

Philip V. Ogren. 2006. Knowtator: A protégé plug-in for annotated corpus construction. In *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Demonstrations*, pages 273–275, New York City, USA. Association for Computational Linguistics.

Gautam Kishore Shahi and Tim A Majchrzak. 2022. Amused: An annotation framework of multimodal social media data. In *International Conference on Intelligent Technologies and Applications*, pages 287–299. Springer.

Kushagra Singh, Indira Sen, and Ponnurangam Kumaraguru. 2018. A Twitter corpus for Hindi-English code mixed POS tagging. In *Proceedings of the Sixth International Workshop on Natural Language Processing for Social Media*, pages 12–17, Melbourne, Australia. Association for Computational Linguistics.

Pontus Stenetorp, Sampo Pyysalo, Goran Topić, Tomoko Ohta, Sophia Ananiadou, and Jun'ichi Tsujii. 2012. brat: a web-based tool for nlp-assisted text annotation. In *Proceedings of the Demonstrations at the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 102–107, Avignon, France. Association for Computational Linguistics.

Jie Yang, Yue Zhang, Linwei Li, and Xingxuan Li. 2018. Yedda: A lightweight collaborative text span annotation tool. *ACL 2018*, page 31.

Seid Muhie Yimam, Iryna Gurevych, Richard Eckart de Castilho, and Chris Biemann. 2013. WebAnno: A flexible, web-based and visually supported system for distributed annotations. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 1–6, Sofia, Bulgaria.

A Appendix

A.1 Inter-annotator agreement (IAA)

IAA measures how well multiple annotators can make the same annotation decision for a particular category. IAA shows you how clear your annotation guidelines are, how uniformly your annotators understand them, and how reproducible the annotation task is. Cohen's kappa coefficient (Hallgren, 2012; Cohen, 1960) is a statistic to measure the reliability between annotators for qualitative (categorical) items. It is a more robust measure than simple percent agreement calculations, as k considers the possibility of the agreement occurring by chance. It is a pairwise reliability measure between two annotators.

The formula for Cohen's kappa (κ) is:

$$\kappa = \frac{P_o - P_e}{1 - P_e} \tag{1}$$

where, P_o is *relative observed agreement among raters* and P_e is *hypothetical probability of chance agreement*.

A.2 Code-mixing Index (CMI)

CMI metric (Das and Gambäck, 2014b) is defined as follows:

$$CMI = \begin{cases} 100 * [1 - \frac{max(w_i)}{n-u}] & n > u \\ 0 & n = u \end{cases} \tag{2}$$

Here, w_i is the number of words of the language i, max$\{w_i\}$ represents the number of words of the most prominent language, n is the total number of tokens, u represents the number of language-independent tokens (such as named entities, abbreviations, mentions, and hashtags). A low CMI score indicates monolingualism in the text whereas the high CMI score indicates the high degree of code-mixing in the text.

B Limitations

We present some of the limitations in the COMMENTATOR tool, along with potential areas for future improvement:

1. **Web-hosting**: COMMENTATOR is not currently web-based, but we are developing a web version to improve accessibility and user experience.

2. **Model Integration**: The tool does not yet support direct integration of pre-trained models through the user interface for predictions.

3. **Post-annotation Analysis**: While offering basic post-annotation analysis, future versions will include task-specific metrics such as Fleiss' Kappa, Krippendorff's Alpha, and Intraclass Correlation for more detailed evaluations of inter-annotator reliability and annotation accuracy.

C Acknowledgements

This work is supported by the Science and Engineering Research Board (SERB) through the project titled "Curating and Constructing Benchmarks and Development of ML Models for Low-Level NLP Tasks in Hindi-English Code-Mixing". The authors express their gratitude to Diksha, Mahesh Kumar, Ronakpuri Goswami, Vaidahi Patel and Ashish Singh for their invaluable support with annotation. We also extend our thanks to Vannsh Jani, Isha Narang, and Eshwar Dhande for their assistance in reviewing the manuscript and reporting on installation and configuration times. Himanshu Beniwal is supported by the Prime Minister Research Fellowship (PMRF ID-1702154).