International Journal of Engineering Trends and Technology
ISSN: 2231-5381/ https://doi.org/10.14445/22315381/IJETT-V74I4P111

Volume 74 Issue 4, 144-154, April 2026

Original Article

A Pre-processing Tool for Management of Aerospace OOV Words for Building a Bilingual Corpus of English to Assamese

Pratul Kalita[1], Saptarshi Paul[2], Bimal Kumar Kalita[3], Saurav Paul[4]

[1,2,4]*Department of Computer Science, Assam University, Assam, India.*
[3]*Department of Computer Application, Assam Don Bosco University, Assam, India.*

[1]*Corresponding Author : pratul.kalita@aus.ac.in*

Received: 03 June 2025 Revised: 22 January 2026 Accepted: 19 February 2026 Published: 29 April 2026

Abstract *- OOV terminology and structured English phrases, which are exclusive to the aerospace industry, are crucial to this field. Translating these words and phrases is a difficult task. The use of social media, such as Facebook, Twitter, etc., by airlines as a significant tool for communication and advertising has led to the inclusion of these aerospace-related terms in everyday reading materials like newspapers. To make it easier for MT(Machine Translation) systems to translate these structured phrases and OOV terms, such as TFC, LEO, a tool that can recognize them and substitute them with the appropriate meaningful English word is urgently needed. For accurate translation, the MT(Machine Translation) systems must then be trained utilizing aerospace parallel corpora (English-target language). Another gap that must be remedied is the lack of such a corpus. There is a great demand for a tool that can handle both the direct feed to MT tools and assist in building a parallel corpus.*

Keywords *- Aerospace, OOV Terms, Machine Translation, Corpus.*

1. Introduction

India is one of the fastest-growing major aviation sectors in the world and the third-largest domestic market. India's aviation industry has experienced significant expansion, as evidenced by the 15% rise in the total number of passengers handled at Indian airports, which reached 37.6 crore in the financial year 2024. The fact that the total revenue of Airport Authority of India (AAI) has increased by 3807.48 crores (31.28%) in the financial year 2023–2024 indicates the direct effect on its performance [1]. With the rapid increase in the number of passengers, the airline companies have to introduce more aircraft to fulfill the demand. At the same time, huge job opportunities have been created. The Indian aircraft industry is one of the sectors experiencing the fastest expansion. India is expected to have the second-largest aerospace industry by 2025. At the moment, India's aerospace and aviation sector exhibits both achievements and challenges. Satellite launches and missions to the moon and Mars are among India's achievements. Prominent organizations like ISRO, HAL, and DRDO make important contributions. However, the industry needs assistance due to the high regulatory requirements, lack of technology, and minimal private sector engagement. In order to become self-sufficient and globally competitive, India seeks to improve R&D, form alliances, and assist entrepreneurs. The nation's aerospace scene demonstrates a range of successes and objectives, suggesting a dynamic course with much room for expansion and innovation [2]. There is a huge need for Machine Translation (MT) systems in the aviation as well as aerospace domains, which can translate sentences of the English language to Indian languages. Already several MT systems were developed in domains like medical, tourism, etc., but few MT systems were built in the aerospace domain [11]. Regional Air Connectivity is envisioned under the RCS (Regional-Connectivity Scheme) in the NCAP, 2016 (National-Civil-Aviation Policy, 2016). On 21 October, 2016, MoCA (The Ministry of Civil Aviation) introduced UDAN (Ude Desh ka Aam Nagrik) under RCS. RCS-UDAN's main goal is to make regional air connections more accessible and inexpensive for the general public [3]. The number of airports has increased concurrently, rising from 74 in 2014 to 147 in 2022. By 2024–2025, MoCA wants to raise this to 220 [4]. The number of airports, as well as job opportunities, has grown increasingly in Assam. As Assamese is the prime language in Assam, there is a great need for an English to Assamese MT system. The aerospace domain broadly uses unique OOV (Out of Vocabulary) words. Translating these words is not an easy task.

2. Literature Review

A Word2VnCR technique was presented by Jeongin Kim et al. (2021) [7] to substitute an OOV word with a term that is semantically related in the event of a morpheme analysis

error. This method extracts candidate words that have the same meaning as the OOV word and can be used to determine how similar they are semantically to the OOV word's neighboring terms. Additionally, they compared the Word2VnCR and Word2Vec algorithms and discovered that Word2VnCR was more effective in substituting semantically similar words for OOV keywords. Using a small bilingual parallel corpus and two unrelated and unlabeled raw corpora for both target and source languages, Vijay et al. (2022) [8] proposed a translation algorithm, which is context-based SMT for OOV words translation. For FIRE 2010 and Fire 2011, the reduction of the OOV terms has been claimed by up to 0.81% and 1.73%, respectively. When the context words list is known, Baby et al. (2022) [16] proposed a post-processing method that recovers OOV terms, which is context-based using the phone cost function. Additionally, they examined cost functions based on phonetic and acoustic similarity. Their findings show that the proposed method offers above 50% OOV context word recovery in a number of categories. According to a review of the literature by us, in the context of Indian languages, comparatively few researches have been done on OOV words. For a low-resource language like Assamese, still, no research work could be found for the translation of OOV terms. Here, in this paper, after translating OOV terms into standard terms, a bilingual corpus of English to Assamese will also be created.

3. OOV Words

Any terms that a machine learning model has not come across during its training phase are referred to as Out-of-Vocabulary (OOV) words in Natural Language Processing (NLP). Processing text or voice that contains these terms is extremely difficult because they are not included in the model's predefined vocabulary. When these models come across an OOV word, it may result in mistakes or inaccuracies because they mostly rely on their taught vocabulary to interpret input [5]. The fact that language is always evolving is one of the main causes of Out-of-Vocabulary (OOV) words. Regional dialects, technical developments, and cultural trends all contribute to the continuous creation of new words. This linguistic growth is accelerated by social media in particular, where hashtags, slang, and abbreviations are regularly added to the language [12].

Additionally, words from various languages, cultures, or geographical areas frequently blend with mainstream languages as global contact grows. Traditional NLP models, which rely on a fixed vocabulary, struggle to keep pace with these changes, leading to OOV words in real-world applications [13]. To detect OOV words, models of word embedding, such as BERT, are used, which are trained. A quite common requirement for every transliteration and transliteration tools is to handle out-of-vocabulary (OOV) words. These are Technical words, person names, names of places, objects, etc., all of which occur frequently in texts, especially those related to aerospace, and for a transliteration system, it is absolutely necessary to identify such OOV words and process them accordingly. [10]. To create the corpus of OOV words is a difficult task. The sources of OOV words, as collected with proper permission as and where necessary, are mentioned below:-

1. Directorate General of Civil Aviation (DGCA)
2. Airport Authority of India (AAI).
3. Indian Space Research Organization (ISRO)
4. Hindustan Aeronautics Limited (HAL)
5. National Aeronautics and Space Administration (NASA)

A parallel corpus of the aerospace domain has been created with a size of 19959 sentences. The corpus contains 359 OOV words. Tables 1 and 2 show some unique Aviation abbreviations, OOV words, and Aerospace OOV words.

Table 1. Unique Aviation abbreviations and OOV words [4]

OOV words	Meaning
CCU	Netaji Subhas Chandra Bose International Airport
GAU	Lokpriya Gopinath Bordoloi International Airport
IXS	Silchar Airport
MAA	Chennai International Airport
FL	Flight Level
TFC	Traffic
DSND	Descend
SBOUND	South Bound
6E	Indigo Airlines
AAI	Airport Authority of India

Table 2. Aerospace OOV words [6]

Purpose	OOV word	Meaning
Flight Operations and Procedures	QNH	• Altimeter Setting (to measure height above sea level)
	VOR	• VHF Omni-Directional Range (radio navigation aid)
Aircraft Systems and Technologies	FMS	• Flight Management System
	FADEC	• Full Authority Digital Engine Control

Air Traffic Management (ATM)	ATIS	• Automatic Terminal Information Service
	NOTAM	• Notice to Airmen (or Air Missions)
Airport and Navigation Codes	ICAO	• International Civil Aviation Organization
	RNAV	• Area Navigation
Emerging Technologies and Concepts	eVTOL	• Electric Vertical Takeoff and Landing
	UTM	• Unmanned Aircraft Systems Traffic Management
Space Technology	LEO	• Low Earth Orbit
	SLV	• Space Launch Vehicle
	ISS	• International Space Station

4. Problem Statement

OOV terms are used extensively in the extremely specialized and technical field of aerospace. The main issue is to find these OOV terms and convert them to proper, pertinent forms before translating them into the goal language. Another difficulty is figuring out which structured phrases are used in social media and aerospace, then replacing them with equivalent English terms before translating. Special characters like @, #, and so on accompany these OOV and structured words, making it impossible for current translators to recognize, comprehend, and convert them. The MT program is unable to comprehend the true meaning of the many aerospace articles and conversations it encounters. At the end, the MT software produces an output that is either easily understood or translated inadequately, leaving the OOV terms untranslated or, at most, transliterated. Since these structured and OOV words are typically not included in the training database, the most frequent issue with MT(Machine Translation) software is its inability to comprehend sentences in the source language. As an illustration: "I am traveling from GAU to CCU Today," says a conversation. This indicates that "I am traveling from Guwahati Lokpriyo Gopinath Bordoloi International Airport to Kolkata Airport today". Using a conventional machine translation program, it has been observed that the aforementioned statement can not be translated or transliterated since it does not comprehend the words "GAU" and "CCU", and it reads as: "আজি মই GAU ৰ পৰা CCU লৈ যাত্রা কৰি আছো."

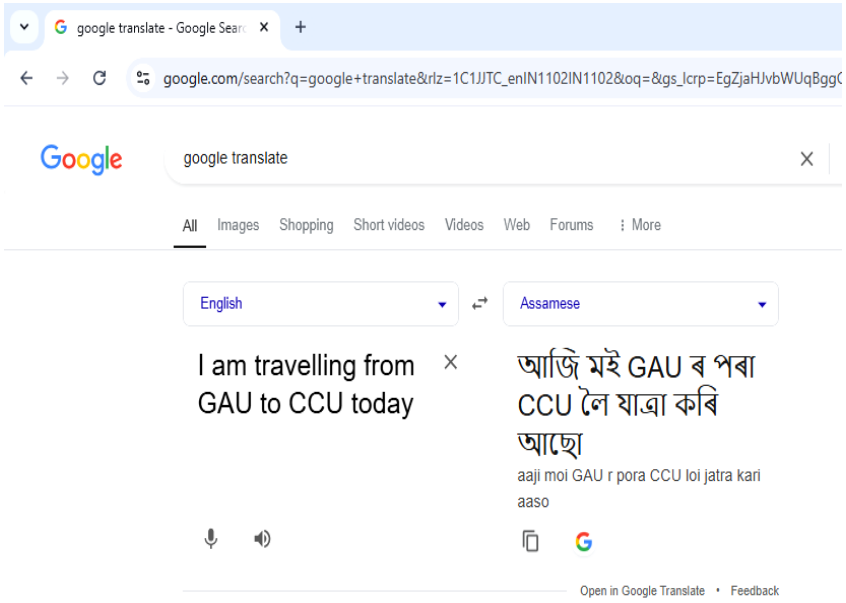


Fig. 1 Inaccurate translation of the sentence by Google Translate

5. The Proposal to Solve the Problem

A straightforward key is to create a tool that converts OOV words, structured words, and special characters contained in sentences into regular English sentences before feeding them into the program, which can translate. This way, the MT program can comprehend the sentences and translate them accurately. In addition to training MT systems to develop models that can handle OOV words and other similar terms, Bilingual Parallel (Source-Target) databases can also be created using this program. Creating a database of these structured and OOV phrases alongwith their definitions, can help to achieve this. When such words appear in a phrase, they can be substituted with the appropriate, meaningful word by consulting this database. The MT(Machine Translation) software can then use this pre-processed text to transform it successfully and productively.

5.1. Methodology

JavaScript programming has been used here to accomplish the goals as follows:

- To swap out the OOV terminology used in aerospace for their common-place equivalents or meanings in ordinary English.
- To recognize and eliminate special characters like @, #, and so forth. The OOV and structured words are kept in a database to which the software makes reference.

Rodali Keyboard has been used to write the Assamese alphabet so as to create the parallel corpus. [14]. A snapshot of the Rodali keyboard is shown in Figure 2.

Fig. 2 Rodali keyboard to write Assamese alphabets [9]

5.2. Functionality of the Program

The application is created using JavaScript to substitute standard English/MT comprehensible vocabulary for the structured and OOV words used in aerospace, making them easier for MT translators to understand. Such organized OOV words are compiled into a database with definitions.

The database is open, and the process of adding new words is ongoing. The delimiter columns (oov_term and replacement) in the database keep words apart from their definitions. Common in-short phrases are replaced with their full counterparts by the pre-process Sentence function. Additionally, the software employs regular expressions inside pre-process Sentences to replace special characters like "#" (hashtags), "@" (at-the-rate), and "_" (under-score).

Using the aforementioned example, an aerospace specialist shared on social media, "I am traveling from GAU to CCU today." The OOV terms "GAU" and "CCU," which stand for Guwahati Lokpriyo Gopinath Bordoloi International Airport and Kolkata Airport, respectively, and are maintained in the database, are substituted by our program as "I am traveling from Guwahati Lokpriyo Gopinath Bordoloi International Airport to Kolkata Airport today." The file name where terms are saved is provided by the sqlite3—database ('./aviation_terms.db') function. Db.all(...) has been used as a "read" operation. The application uses the /pre-process route to receive the input data, which includes hashtags, underscores, and other information. The hashtags are removed by the pre-process Sentence function (#). The pre-process Sentence function is used to substitute the OOV and structured phrases with the words that are meaningful and machine-translatable. The JavaScript application incorporates the sqlite3 library to convert the terms to their standard definitions. Tokens are created by splitting the string using the split(/\s+/) function. The program has been used to attain a straightforward solution. The forms of noise (OOV terms, special characters) can be identified with the aid of a manually constructed database, which can then be replaced with words that are meaningful, and then they are easy for MT software to translate.

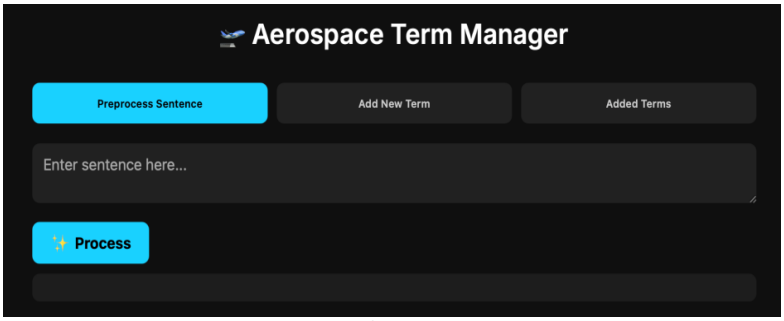


Fig. 3 Homepage of the pre-processing tool

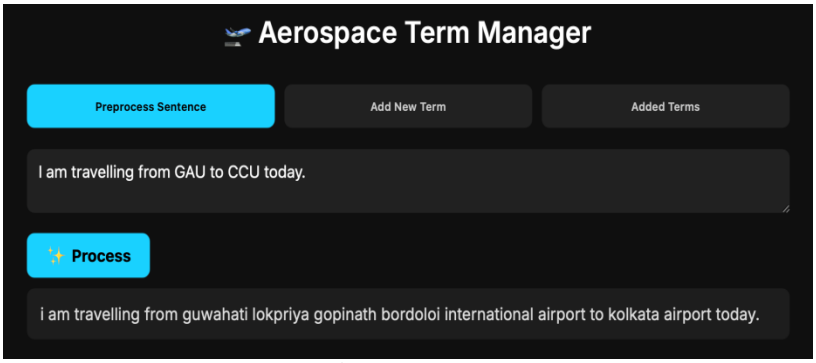


Fig. 4 Translation of a sentence containing OOV words

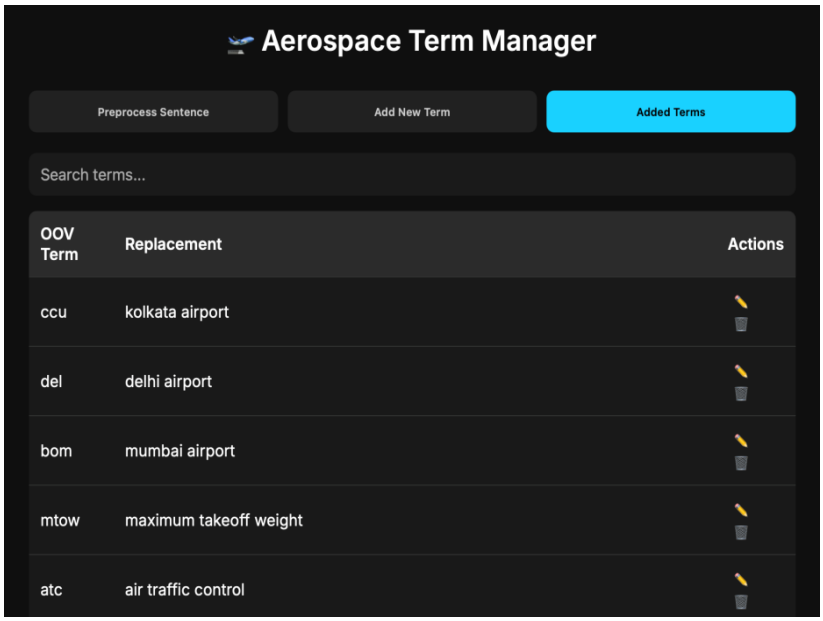


Fig. 5 Page showing the corresponding terms of OOV words

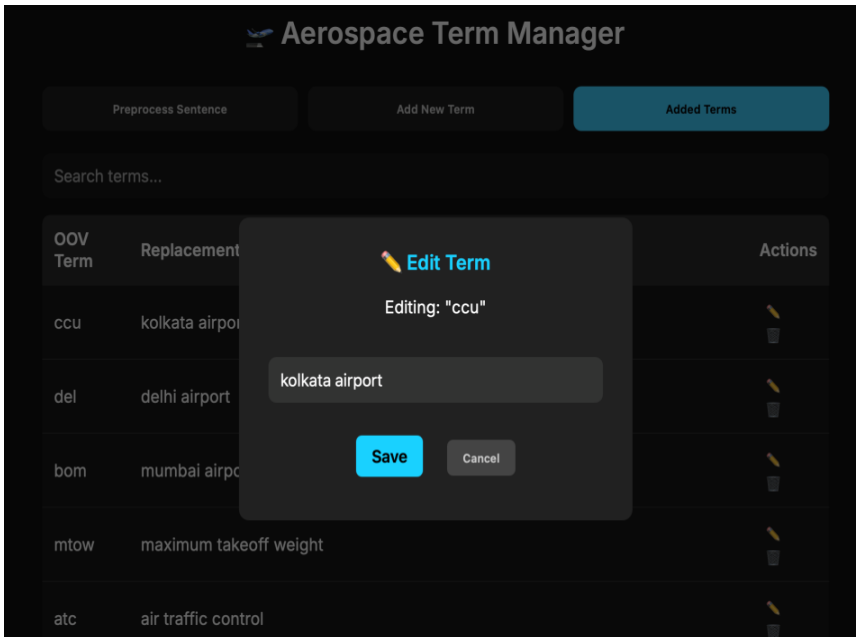


Fig. 6 Edit or Delete form

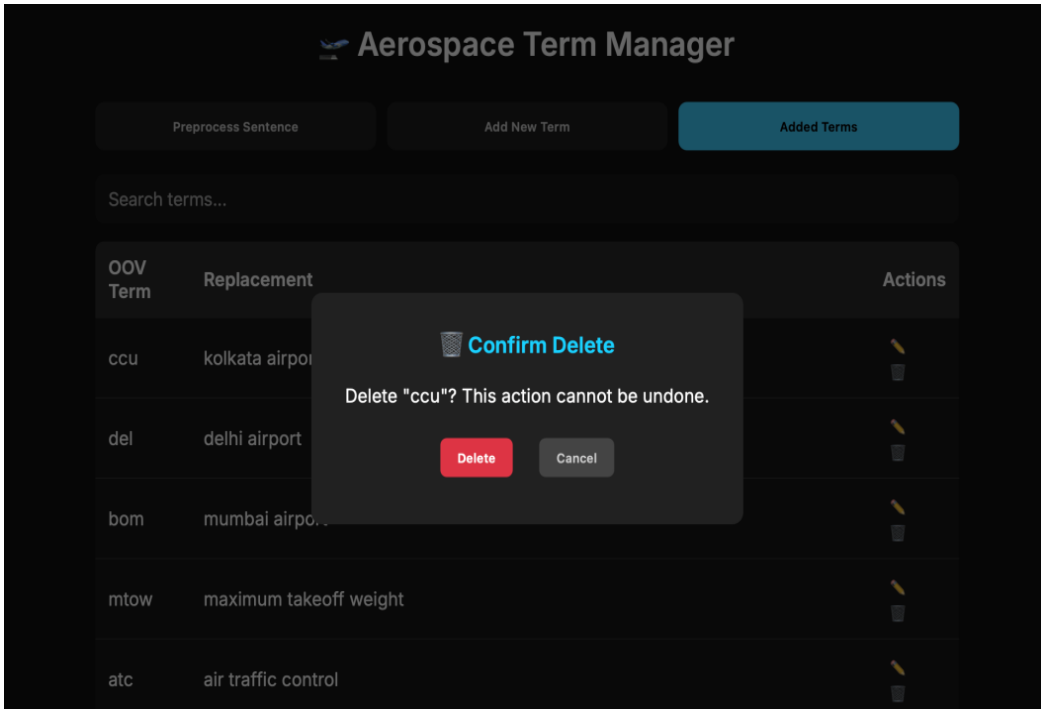


Fig. 7 Edit or Delete confirmation message

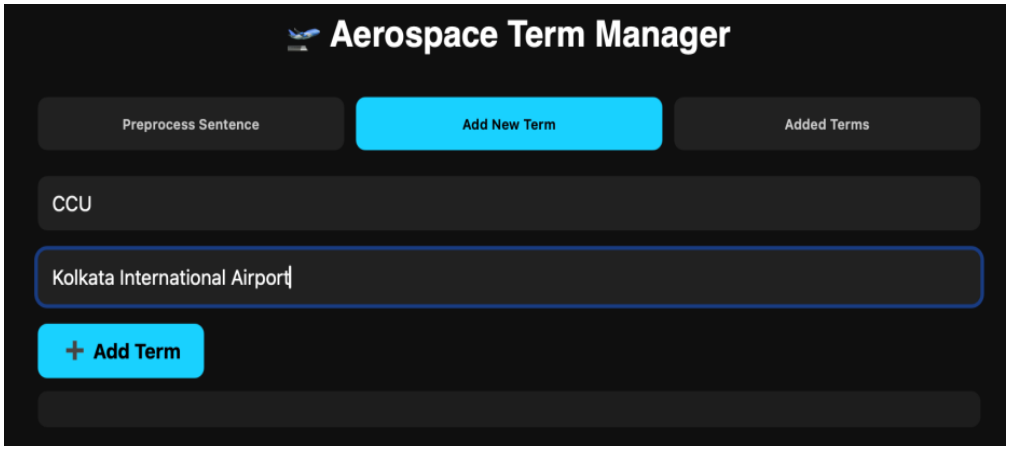


Fig. 8 Adding new terms

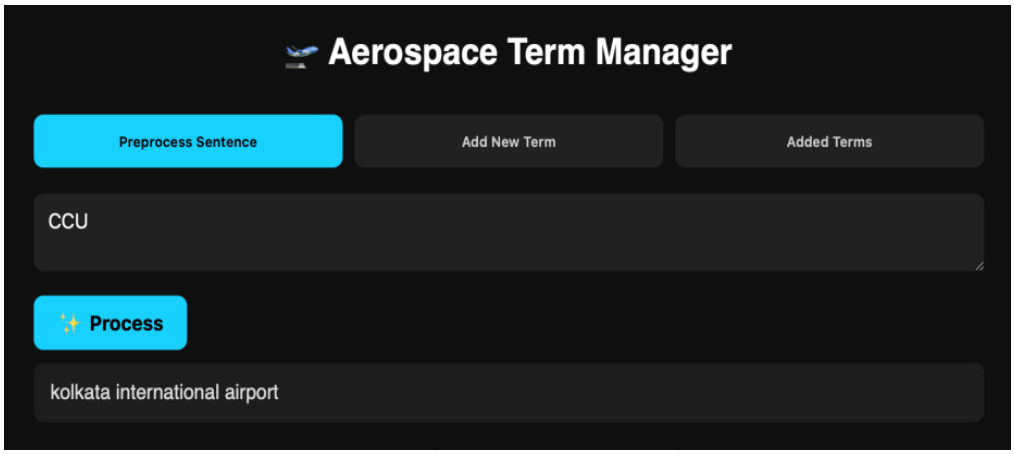


Fig. 9 After the addition of the corresponding term of the OOV word

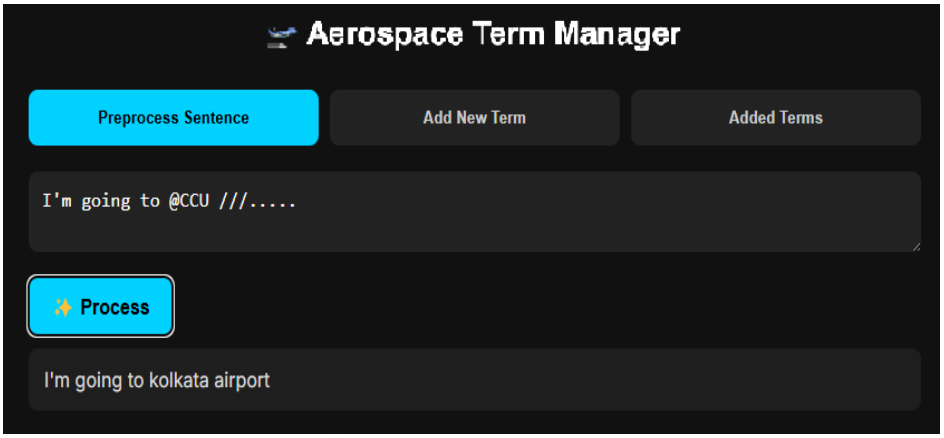


Fig. 10 Processing of a sentence containing a special character (e.g., @)

```
// ✅ DELETE: Remove a term
app.delete('/delete-term/:oov_term', (req, res) => {
  const termToDelete = req.params.oov_term.toLowerCase().trim();

  db.run(
    'DELETE FROM term_mappings WHERE oov_term = ?',
    [termToDelete],
    function(err) {
      if (err) {
        console.error('Delete error:', err.message);
        return res.status(500).json({ error: 'Failed to delete term' });
      }

      if (this.changes === 0) {
        return res.status(404).json({ error: 'Term not found' });
      }

      res.json({
        message: 'Term deleted successfully',
        deletedTerm: termToDelete
      });
    }
  );
});

// Start server
app.listen(port, () => {
  console.log(`✅ Server running at http://localhost:${port}`);
  console.log(`📘 Swagger UI available at http://localhost:${port}/api-docs`);
});
```

Fig. 11 Code for Deletion

```
// ✅ GET: All terms
app.get('/terms', (req, res) => {
  db.all('SELECT oov_term, replacement FROM term_mappings', (err, rows) => {
    if (err) {
      console.error('Failed to retrieve terms:', err.message);
      return res.status(500).json({ error: 'Failed to fetch terms' });
    }
    res.json({ count: rows.length, terms: rows });
  });
});

// ✅ PUT: Edit a term
app.put('/edit-term/:oov_term', (req, res) => {
  const originalTerm = req.params.oov_term.toLowerCase().trim();
  const { replacement } = req.body;

  if (!replacement || typeof replacement !== 'string') {
    return res.status(400).json({ error: 'Replacement must be a valid string' });
  }

  const cleanReplacement = replacement.toLowerCase().trim();

  db.run(
    'UPDATE term_mappings SET replacement = ? WHERE oov_term = ?',
    [cleanReplacement, originalTerm],
    function(err) {
      if (err) {
        console.error('Update error:', err.message);
        return res.status(500).json({ error: 'Failed to update term' });
      }

      if (this.changes === 0) {
        return res.status(404).json({ error: 'Term not found' });
      }

      res.json({
        message: 'Term updated successfully',
        term: originalTerm,
        newReplacement: cleanReplacement
      });
    }
  );
});
```

Fig. 12 Code for Edit

```
// POST: Preprocess sentence
app.post('/preprocess', async (req, res) => {
  try {
    const { sentence } = req.body;
    if (!sentence || typeof sentence !== 'string') {
      return res.status(400).json({ error: 'Please provide a valid sentence' });
    }
    const processedSentence = await preprocessSentence(sentence);
    res.json({ processed: processedSentence });
  } catch (error) {
    console.error('Error processing sentence:', error);
    res.status(500).json({ error: 'Internal server error' });
  }
});

// POST: Add OOV term
app.post('/add-term', (req, res) => {
  const { oov_term, replacement } = req.body;
  if (!oov_term || !replacement || typeof oov_term !== 'string' || typeof replacement !== 'string') {
    return res.status(400).json({ error: 'Please provide valid oov_term and replacement as strings' });
  }

  const cleanOovTerm = oov_term.toLowerCase().trim();
  const cleanReplacement = replacement.toLowerCase().trim();

  if (!cleanOovTerm || !cleanReplacement) {
    return res.status(400).json({ error: 'Terms cannot be empty after trimming' });
  }

  db.run(
    'INSERT OR REPLACE INTO term_mappings (oov_term, replacement) VALUES (?, ?)',
    [cleanOovTerm, cleanReplacement],
    function(err) {
      if (err) {
        console.error('Database error:', err.message);
        return res.status(500).json({ error: 'Failed to add term: ' + err.message });
      }
      res.json({
        message: 'Term added successfully',
        term: cleanOovTerm,
        replacement: cleanReplacement
      });
    }
  );
});
```

Fig. 13 Code for Addition

```
const express = require('express');
const sqlite3 = require('sqlite3').verbose();
const swaggerUi = require('swagger-ui-express');
const YAML = require('yamljs');
const cors = require('cors');

const swaggerDocument = YAML.load('./swagger.yaml');
const app = express();
const port = 3000;

// Middleware
app.use(express.json());
app.use(cors());
app.use('/api-docs', swaggerUi.serve, swaggerUi.setup(swaggerDocument));

// Connect to DB
const db = new sqlite3.Database('./aviation_terms.db', (err) => {
  if (err) {
    console.error('Database connection error:', err.message);
  } else {
    console.log('Connected to SQLite database');
  }
});

// Preprocessing
async function preprocessSentence(sentence) {
  return new Promise((resolve, reject) => {
    let words = sentence.toLowerCase().split(/\s+/);
    db.all('SELECT oov_term, replacement FROM term_mappings', (err, rows) => {
      if (err) return reject(err);
      const termMap = new Map(rows.map(row => [row.oov_term, row.replacement]));
      words = words.map(word => {
        const cleanWord = word.replace(/[.,!?]/g, '');
        return termMap.has(cleanWord) ? termMap.get(cleanWord) : word;
      });
      resolve(words.join(' '));
    });
  });
}
```

Fig. 14 Code for Pre-processing

5.3. Flowchart of the Program

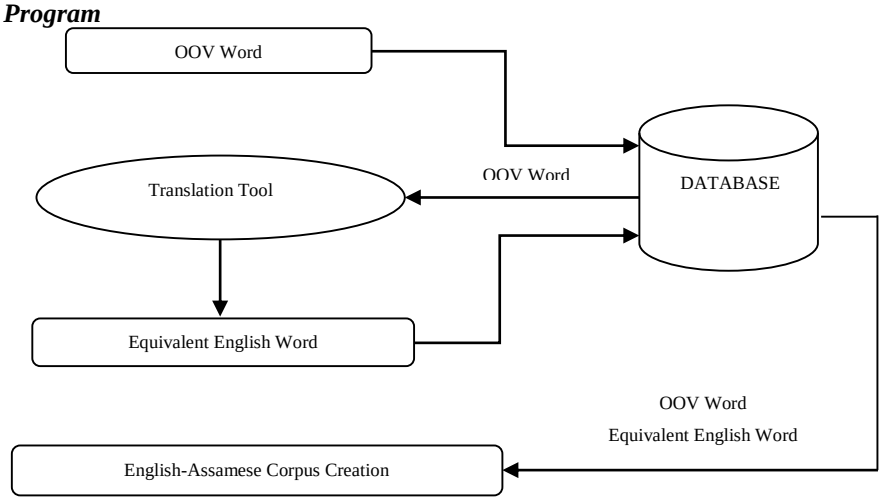


Fig. 15 Flowchart of the tool

6. Results and output

6.1. Machine Translation Software as a pre-Processing Tool to Input Directly

Referring back to the earlier example in Figure 1, we enter the identical statement into our suggested software, and the result is: “Today, I am traveling from Guwahati Lokpriyo Gopinath Bordoloi International Airport to Kolkata Airport.” When we supply the above output obtained into Google Translate, the translation appears as follows in Figure 16. In this case, Google Translate successfully translated the statement (as shown in Figure 16) from English to Assamese while preserving its meaning, which is:

“আজি গুৱাহাটী লোকপ্ৰিয় গোপীনাথ বৰদলৈ আন্তঃৰাষ্ট্ৰীয় বিমানবন্দৰৰ পৰা কলকাতা বিমানবন্দৰলৈ যাত্ৰা কৰি আছোঁ।”

The ability of this tool to assist in the creation of a bilingual Source to target language, i.e., English to Assamese corpus for the aerospace domain, is one of its strengths. In order to get precise findings from commercially accessible MT systems, it may be utilized as a pre-processing tool.

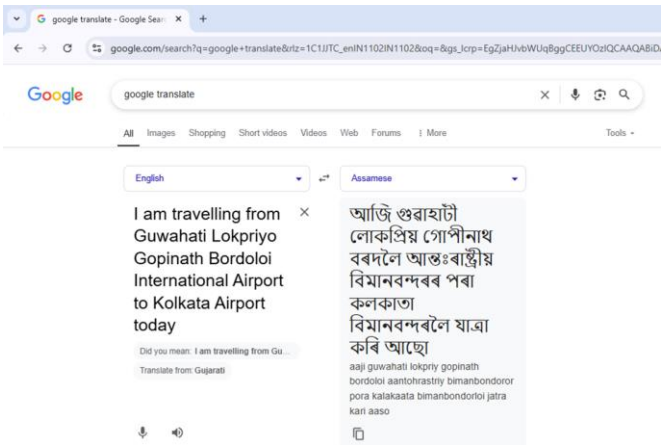
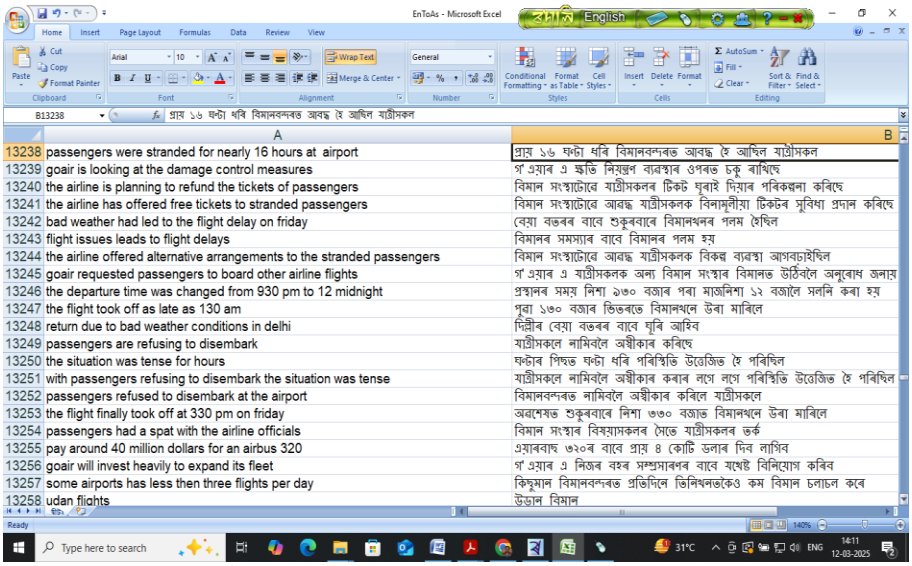


Fig. 16 Accurate translation provided by suggested software using Google Translate

6.2. To Create Bilingual Corpus

Any bilingual corpus (English-target language) can be created with the help of the tool. It is currently being used to develop English-to-Assamese corpora for the aerospace domain. Regular English sentences are obtained by directly feeding the tool with aerospace data and sentences gathered from multiple sources. These sentences are used to act as the source language. Then, relevant sentences of the target language in Assamese are constructed and inserted into the relevant column. Any other target language can be added to this application. Figure 17 shows a snapshot of the corpus to illustrate how the software can be used as a tool to help create corpora. Any SMT or NMT system can be trained using this bilingual corpus. [10]



	A	B
13238	passengers were stranded for nearly 16 hours at airport	প্ৰায় ১৬ ঘণ্টা ধৰি বিমানবন্দৰত আবদ্ধ হৈ আছিল যাত্ৰীসকল
13239	goair is looking at the damage control measures	গ’ এয়াৰ এ ক্ষতি নিয়ন্ত্ৰণ ব্যৱস্থাৰ ওপৰত চকু ৰাখিছে
13240	the airline is planning to refund the tickets of passengers	বিমান সংস্থাটোৱে যাত্ৰীসকলৰ টিকট ঘুৰাই দিয়াৰ পৰিকল্পনা কৰিছে
13241	the airline has offered free tickets to stranded passengers	বিমান সংস্থাটোৱে আবদ্ধ যাত্ৰীসকলক বিনামূলীয়া টিকটৰ সুবিধা প্ৰদান কৰিছে
13242	bad weather had led to the flight delay on friday	বেয়া বতৰৰ বাবে শুকুৱবাৰে বিমানখনৰ পলম হৈছিল
13243	flight issues leads to flight delays	বিমানৰ সমস্যাৰ বাবে বিমানৰ পলম হয়
13244	the airline offered alternative arrangements to the stranded passengers	বিমান সংস্থাটোৱে আবদ্ধ যাত্ৰীসকলক বিকল্প ব্যৱস্থা আগবঢ়াইছিল
13245	goair requested passengers to board other airline flights	গ’ এয়াৰ এ যাত্ৰীসকলক অন্য বিমান সংস্থাৰ বিমানত উঠিবলৈ অনুৰোধ জনায়
13246	the departure time was changed from 930 pm to 12 midnight	প্ৰস্থানৰ সময় নিশা ৯৩০ বজাৰ পৰা মাজনিশা ১২ বজালৈ সলনি কৰা হয়
13247	the flight took off as late as 130 am	পুৱা ১৩০ বজাৰ ভিতৰতে বিমানখনে উৰা মাৰিলে
13248	return due to bad weather conditions in delhi	দিল্লীৰ বেয়া বতৰৰ বাবে ঘূৰি আহিব
13249	passengers are refusing to disembark	যাত্ৰীসকলে নামিবলৈ অস্বীকাৰ কৰিছে
13250	the situation was tense for hours	ঘণ্টাৰ পিছত ঘণ্টা ধৰি পৰিস্থিতি উত্তেজিত হৈ পৰিছিল
13251	with passengers refusing to disembark the situation was tense	যাত্ৰীসকলে নামিবলৈ অস্বীকাৰ কৰাৰ লগে লগে পৰিস্থিতি উত্তেজিত হৈ পৰিছিল
13252	passengers refused to disembark at the airport	বিমানবন্দৰত নামিবলৈ অস্বীকাৰ কৰিলে যাত্ৰীসকলে
13253	the flight finally took off at 330 pm on friday	অৱশেষত শুকুৱবাৰে নিশা ৩৩০ বজাত বিমানখনে উৰা মাৰিলে
13254	passengers had a spat with the airline officials	বিমান সংস্থাৰ বিষয়াসকলৰ সৈতে যাত্ৰীসকলৰ তৰ্ক
13255	pay around 40 million dollars for an airbus 320	এয়াৰবাছ ৩২০ৰ বাবে প্ৰায় ৪ কোটি ডলাৰ দিব লাগিব
13256	goair will invest heavily to expand its fleet	গ’ এয়াৰ এ নিজৰ বহৰ সম্প্ৰসাৰণৰ বাবে যথেষ্ট বিনিয়োগ কৰিব
13257	some airports has less then three flights per day	কিছুমান বিমানবন্দৰত প্ৰতিদিনে তিনিখনতকৈও কম বিমান চলাচল কৰে
13258	udan flights	উড়ান বিমান

Fig. 17 Snapshot of the English-Assamese Aerospace Corpus created using the pre-processing tool

7. Conclusion

It makes sense to pre-process source language sentences to standardize them before feeding them into an MT system for direct translations, especially in technical domains like aerospace. Few pre-processing software can be found for specialized domains like aviation and aerospace. The aforementioned tools can help to create parallel bilingual corpora in English as the source language for any target language. It is an independent application that might be applied to generate English sentences in the aerospace field before they are fed to any common readymade translation software to improve translation quality, in addition to being a pre-processing tool for corpus building.

7.1. Future work

The database is active and can occasionally be added to as new aerospace OOV phrases as and when acronyms and technical jargon are discovered. Since it is a technical corpus, it can be combined with a sizable portion of general English to the Assamese parallel corpus to train NMT software like OpenNMT and generate models that can be used for translation and transliteration once it has a significant number of OOV and technical words.

References

[1] Airport Authority of India (AAI), 2026. [Online]. Available: https://www.aai.aero/

[2] All India Association of Industries (AIAI), Aerospace, 2026. [Online]. Available: https://aiaiindia.com/aerospace/

[3] Airport Authority of India (AAI), National Civil Aviation Policy 2016. [Online]. Available: https://www.aai.aero/en/node/4528

[4] MoCA-Ministry of Civil Aviation, 2026. [Online]. Available: https://www.civilaviation.gov.in/

[5] Zhongyu Zhuang et al., "Out-of-Vocabulary Word Embedding Learning based on Reading Comprehension Mechanism," *Natural Language Processing Journal*, vol. 5, pp. 1-6, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[6] Gregory S. Jones et al., Research Opportunities Aerospace Concepts, Hampton, Virginia: National Aeronautics and Space Administration (NASA), Langley Research Center, 2000. [Online]. Available: https://searchworks.stanford.edu/view/12264135

[7] Jeongin Kim, Taekeun Hong, and Pankoo Kim, "Replacing out-of-Vocabulary Words with an Appropriate Synonym based on Word2VnCR," *Mobile Information Systems*, vol. 2021, no. 1, pp. 1-7, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[8] Vijay Kumar Sharma, Namita Mittal, and Ankit Vidyarthi, "Context-based Translation for the Out of Vocabulary Words Applied to Hindi-English Cross-Lingual Information Retrieval," *IETE Technical Review*, vol. 39, no. 2, pp. 276-285, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[9] Rodali Assamese Keyboard. [Online]. Available: https://rodali-assamese-keyboard.en.softonic.com/android

[10] Sahinur Rahman Laskar et al., "Improving English-Assamese Neural Machine Translation using Transliteration-based Approach," *Evolution in Computational Intelligence: Proceedings of the 10th International Conference on Frontiers in Intelligent Computing: Theory and Applications*, Cardiff, United Kingdom, vol. 326, pp. 223-231, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[11] Rudolf A. Braun, Srikanth Madikeri, and Petr Motlicek, "A Comparison of Methods for OOV-Word Recognition on a New Public Dataset," *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Toronto, ON, Canada, pp. 5979-5983, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[12] Sahinur Rahman Laskar, Partha Pakray, and Sivaji Bandyopadhyay, "Neural Machine Translation for Low Resource Assamese-English," *Proceedings of the International Conference on Computing and Communication Systems: I3CS 2020*, NEHU, Shillong, India, vol. 170, pp. 35-44, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[13] Mazida Akhtara Ahmed, Kishore Kashyap, and Shikhar Kumar Sarma, "Tokenization Effect on Neural Machine Translation: An Experimental Investigation for English-Assamese," *2023 14th International Conference on Computing Communication and Networking Technologies (ICCCNT)*, Delhi, India, pp. 1-7, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[14] Sahinur Rahman Laskar et al., "EnAsCorp1.0: English-Assamese Corpus," *Proceedings of the 3rd Workshop on Technologies for MT of Low Resource Languages*, Suzhou, China, pp. 62-68, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[15] Sahinur Rahman-Laskar et al., "A Domain Specific Parallel Corpus and Enhanced English-Assamese Neural Machine Translation," *Computación y Sistemas*, vol. 26, no. 4, pp. 1669-1687, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[16] Arun Baby et al., "Context-based out-of-Vocabulary Word Recovery for ASR Systems in Indian Languages," *arXiv preprint*, pp. 1-12, 2022. [CrossRef] [Google Scholar] [Publisher Link]