```
# -*- coding: utf-8 -*-
"""PREPROCESS

Automatically generated by Colab.

Original file is located at
    https://colab.research.google.com/drive/1ZLxEAluaeC3atSWOOm5r0naJFOLDN9VQ
"""

import re
import csv
import pandas as pd
import multiprocessing as mp
import os

# Define the input file path
input_file = '/content/india-news-headlines (1).csv'

# Generate the output file path by appending '_processed' to the input file name
base, ext = os.path.splitext(input_file)
output_file = f"{base}_processed{ext}"

# Pre-compile regex patterns and define stopwords
token_pattern = re.compile(r'\b\w+\b')
stop_words = set([
    'the', 'and', 'is', 'in', 'to', 'with', 'a', 'for', 'of', 'on', 'at', 'by',
'an'
])

# Preprocessing function for a batch of lines
def preprocess_batch(batch):
    processed_batch = []
    for _, row in batch.iterrows():
        text = row['headline_text'].lower()
        tokens = token_pattern.findall(text)
        tokens = [token for token in tokens if token not in stop_words]
        processed_batch.append({
            'publish_date': row['publish_date'],
            'headline_category': row['headline_category'],
            'preprocessed_text': ' '.join(tokens)
        })
    return processed_batch

# Function to process the file in batches with multiprocessing
def process_file_in_batches(input_file, output_file, batch_size=50000):
    with open(output_file, 'w', encoding='utf-8', newline='') as csvfile_out:
        fieldnames = ['publish_date', 'headline_category', 'preprocessed_text']
        writer = csv.DictWriter(csvfile_out, fieldnames=fieldnames)
        writer.writeheader()

        pool = mp.Pool(mp.cpu_count())

        for chunk in pd.read_csv(input_file, chunksize=batch_size,
engine='python'):
            # Process the current batch in parallel
            processed_batch = pool.map(preprocess_batch, [chunk])

            # Flatten the list of processed rows
            for processed_rows in processed_batch:
```

```
                for row in processed_rows:
                    writer.writerow(row)

        pool.close()
        pool.join()

# Run the processing function
process_file_in_batches(input_file, output_file)
```