IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
Impact Factor: 7.301
Volume 3, Issue 5, April 2023

# Extracting Entities from Video and Tagging them - NLP Model

**Anshika Pareek[1], Aditi Mohan[2], Naveen Rathi[3]**
UG Students, Department of Computer Science & Engineering[1,2]
Assistant Professor, Department of Computer Science & Information Technology[3]
Dronacharya College of Engineering, Gurugram, Haryana, India

**Abstract:** *In order to give consumers access to all the crucial details or the excitement of the films, video summarising seeks to create a high-quality text-based summary of the videos. In order to summarise a video, audio files must first be converted to text files, and then the video files. This whole Natural Language Processing's transformer architecture is used in conjunction with the process. Despite the fact that there have been numerous studies on text summarising, we propose our model, an extractive-video summarizer, which is based on cutting-edge pre-trained ML models and open-source libraries. The following regime is utilised by the extractive video summarizer: Creating a transdisciplinary collection of movies, extracting audio from video files, and creating text from audio recordings are just a few examples. the text summarization using (V)Entity extraction and extractive summarizers. Hindi and English, two widely spoken languages in India, were the main subjects of our research. To sum up, our model performs remarkably well and produces appropriate tags for videos.*

**Keywords:** Transformers, Abstractive Summarization, Extractive Summarization, Machine Learning, Natural Language Processing, Video Summarization

## I. INTRODUCTION

It is possible for computers to recognise spoken language and convert it into text thanks to the approaches and technologies developed in the interdisciplinary field of speech recognition in computer science and computational linguistics.

It is also known as speech to text (STT), computer voice recognition, and automatic speech recognition (ASR). Deep learning is widely used in this field, with popular examples including Alexa, Cortana, Google Assistant, and Siri. The way individuals interact with their devices, houses, cars, and employment has altered as a result.

We can speak with a computer or other piece of equipment that can understand what we say and follow our directions thanks to technology. Speech recognition software works by dissecting the audio of a speech recording into individual sounds, assessing each sound, and then transcribing those sounds into text using algorithms to determine the most likely words that belong in that language.

## II. ENTITY EXTRACTION

Machines can automatically recognise or extract entities, such as a product name, event, or place, thanks to entity extraction, also known as named entity extraction (NER). Search engines use it to comprehend user searches, chatbots use it to communicate with people, and teams use it to automate time-consuming chores like data entry.

Entity extraction is a text analysis technique that employs Natural Language Processing (NLP) to extract particular data from text automatically and may classify it in accordance with specified categories. These words or phrases are named entities. In addition to proper names, this also applies to numerical expressions of time or quantity, such as dates, phone numbers, or monetary amounts.

## III. OBJECTIVE

The model's objective is to take the input audio and extract the entities in order to produce an audio file from a video. Applications must process these movies using NLP techniques in order to create audio files and text transcripts from the

IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
Impact Factor: 7.301
Volume 3, Issue 5, April 2023

audio file and its related entities. Extract entities from this text, then add pertinent tags to the videos.The content recommendations will be enhanced using this information. The project's ultimate objective is to fulfil each of the following use cases.

Use Case 1: Transform audio from a video into a radio or podcast.

Audio to audio summarising using Use Case 1a: Audio briefs Case 1b: Text underneath the video with audio converted to text

Use Case 1c: Text to entity extraction from converted data enhances video searchability.

## II. MATERIALS AND METHOD

These are the following APIs and models we have used :

- Moviepy module of Python.
- Speech Recognition Library.
- Bert Extractive Summarizer (Transformer Model).
- Spacy Library for Entity Extraction.

**Procedure**

1. Creating the Dataset.
2. Extraction of Audio from Video.
3. Generating Text from Audio file.
4. Text Summarization.
5. Entity Extraction.

**Creating the Dataset:**

Training set: 25 videos between three and five minutes long.

Test dataset: 20 videos between five and ten minutes long.

Every sort of video is included in the dataset, including discussions on the news, sports commentary, comedic videos, TV episodes, etc.

**Extraction of Audio from Video:**

We are saving the extracted audio in the.wav format (extension) and using the open-source Moviepy library, which produces good results and can accurately convert any sort of video into audio. After gathering the project's dataset, the following step is to turn the videos into audio. We made advantage of the Python Moviepy module to complete that work. We may import this open source library right into our notebook. The outcomes are positive. This library can accurately convert any kind of video to audio. Since.wav is the primary format for uncompressed audio on Microsoft Windows PCs, we are saving that audio in that format.

```
import speech_recognition as sr
import os
from pydub import AudioSegment
from pydub.silence import split_on_silence
import moviepy.editor as mp
```

```
c:\users\asus\appdata\local\programs\python\python39\lib\site-packages\pydub\utils.py:170: RuntimeWarning: Couldn't find ffmpeg or avconv - defaulti
ng to ffmpeg, but may not work
  warn("Couldn't find ffmpeg or avconv - defaulting to ffmpeg, but may not work", RuntimeWarning)
```

```
clip = mp.VideoFileClip(r"work sde.3gpp")
```

```
clip.audio.write_audiofile(r"converted.wav")
```

```
MoviePy - Writing audio in converted.wav

MoviePy - Done.
```

IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
Impact Factor: 7.301
Volume 3, Issue 5, April 2023

**Generating Text from Audio file.**

With the use of speech recognition, we turned audio into text.

With the help of its convenient AudioFile class, SpeechRecognition makes working with audio files simple. This class offers a context manager interface for reading and interacting with the contents of an audio file and can be initialised with the path of an audio file.

API used: - Google Speech Recognition

Splitting long audio files into smaller parts and applying speech recognition to each one of them allows text to be extracted from quiet that lasts 500 milliseconds or longer. Where there is silence for 500 milliseconds or more, audio is split.

**Text Summarization.**

Thus extracted text from the movie is fed through a summarizer to produce a summary. There are essentially two categories of summarizers: extractive and abstractive.

Traditional summary techniques include extractive summarization, in which the gist is created by selecting a few randomly chosen significant sentences from the source text.

In contrast, Abstractive handles summaries in a more sophisticated manner. Here, the summarizer highlights key passages, analyses the surrounding context, and creates a fresh summary. It also makes sure the essential information is communicated. It is therefore a generative way.

Our goal is to create a text summarizer whose input is a lengthy string of words (in a text body) and whose output is a concise summary that is also a string.

We have summarized the text using BERT-extractive-summarizer .

Steps like stop word removal, stemming, and lower-case transformations are purposefully skipped by BERT because it is capable of parsing meaning from all linguistic idiosyncrasies.

The algorithm used for the extractive labelling of the dataset rates the article strategy on a continuum before binning it in the appropriate category. Not every sentence in a summary will have an exact match in the original text because summaries are not completely extractive.

Consequently, we may represent this as a many-to-many seq2seq problem. A Seq2Seq model primarily consists of two elements:

**Encoder**

a collection of many recurrent units (LSTM or GRU cells for higher performance), each of which accepts a single input sequence element and propagates information for that element forward.

In a problem involving question-answering, the input sequence consists of the entire question's words. Each word is denoted by the symbol x_i, where i denotes the word's order.

**Vector Encoder**

This is the model's final hidden state created by the encoder. The formula above is used to calculate it.

In order to aid the decoder in producing precise predictions, this vector seeks to include the data for all input elements. It serves as the decoder portion of the model's first hidden state.

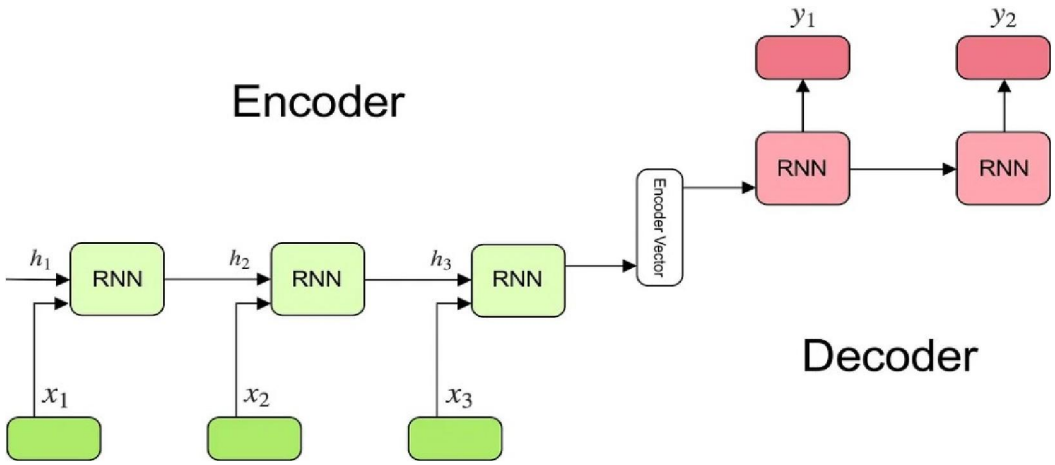
**Decoder**

It is a collection of several recurrent units, each of which forecasts an output at time step t, y_t.

Each recurrent unit receives a hidden state from the prior unit and generates both an output and a hidden state of its own.

The output sequence in the question-answering problem is a compilation of each word in the response. Each word is denoted by the symbol y_i, where i denotes the word's order.

IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
Impact Factor: 7.301
Volume 3, Issue 5, April 2023




Source: https://towardsdatascience.com/understanding-encoder-decoder-sequence-to-sequence-model-679e04af4346

```
print(text)
```

```
There are broadly two types of extractive summarization tasks depending on what the summarization program focuse
tion, which focuses on obtaining a generic summary or abstract of the collection (whether documents, or sets of
c.). The second is query relevant summarization, sometimes called query-based summarization, which summarizes ob
tion systems are able to create both query relevant text summaries and generic machine-generated summaries depen
An example of a summarization problem is document summarization, which attempts to automatically produce an abst
s one might be interested in generating a summary from a single source document, while others can use multiple s
ter of articles on the same topic). This problem is called multi-document summarization. A related application i
a system, which automatically pulls together news articles on a given topic (from the web), and concisely repres
Image collection summarization is another application example of automatic summarization. It consists in selecti
om a larger set of images.[3] A summary in this context is useful to show the most representative images of resu
on system. Video summarization is a related domain, where the system automatically creates a trailer of a long v
consumer or personal videos, where one might want to skip the boring or repetitive actions. Similarly, in survei
ract important and suspicious activity, while ignoring all the boring and redundant frames captured.
```

```
print(summary)
```

```
An example of a summarization problem is document summarization, which attempts to automatically produce an abst
t is generic summarization, which focuses on obtaining a generic summary or abstract of the collection (whether
eos, news stories etc.). The second is query relevant summarization, sometimes called query-based summarization,
o a query. Summarization systems are able to create both query relevant text summaries and generic machine-gener
```

**Entity Extraction.**

Here, entity extraction models are used to extract all the entities from the condensed text and provide tags to the model. We employ Python's spacy library.

Open-source libraries produce good outcomes when used to create systems for information extraction or natural language processing.

Name Entity detection typically includes names of public figures, celebrities, well-known places, and well-known businesses.

The following are a few of the most widely used entity extraction uses:

1. Discover Useful Insights from Customer Feedback

Finding out what your business is doing well and what needs improvement requires that you pay close attention to consumer input. You may discover how consumers feel about your brand and products by looking at survey results, product reviews, or social media posts.

You may quickly find out when customers reference your brand or that of another using entity extraction. For instance, client feedback referencing your rivals and certain products pertinent to your organisation may be of interest to you.

Your clients could wish you had an integration with another service. or a quality that they consistently praise and value

IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
Impact Factor: 7.301
Volume 3, Issue 5, April 2023

2. Get the Information You Need from Support Ticket Data

Managing customer support tickets becomes increasingly difficult as your firm grows. You can manage and arrange all of your tickets in one location by using support desk software like Zendesk. But what about understanding them?

Ticket marking and routing to the appropriate agent are both automated with the aid of entity extraction tools. Use it to retrieve pertinent data from your tickets, such as the type of goods, the date of shipping, and the serial numbers.

Additionally, you can extract corporate names, emails, or URLs and utilise this data to classify and route tickets appropriately.

3. Enhance Content Suggestions

Automating personalised recommendations is one of the entity extraction's many application cases. Large businesses like Amazon and Netflix use extraction techniques to give material that is customised to the likes and behaviour of their customers, keeping them interested all the time.

Recommendation systems can find items with comparable entities by automatically detecting the entities in a product description. This is applicable to a wide range of sectors, including news media and e-commerce.

| TYPE | DESCRIPTION |
|---|---|
| PERSON | People, including fictional. |
| NORP | Nationalities or religious or political groups. |
| FAC | Buildings, airports, highways, bridges, etc. |
| ORG | Companies, agencies, institutions, etc. |
| GPE | Countries, cities, states. |
| LOC | Non-GPE locations, mountain ranges, bodies |
| PRODUCT | Objects, vehicles, foods, etc. (Not services.) |
| EVENT | Named hurricanes, battles, wars, sports event |
| WORK_OF_ART | Titles of books, songs, etc. |
| LAW | Named documents made into laws. |
| LANGUAGE | Any named language. |
| DATE | Absolute or relative dates or periods. |
| TIME | Times smaller than a day. |
| PERCENT | Percentage, including "%". |
| MONEY | Monetary values, including unit. |
| QUANTITY | Measurements, as of weight or distance. |
| ORDINAL | "first", "second", etc. |
| CARDINAL | Numerals that do not fall under another type |

Source: https://towardsdatascience.com/named-entity-recognition-with-nltk-and-spacy-8c4a7d88e7da

OntoNotes 5 corpus was used to train SpaCy's named entity recognition, which covers the above entity types. Token

| TAG | DESCRIPTION |
|---|---|
| B EGIN | The first token of a multi-token entity. |
| I N | An inner token of a multi-token entity. |
| L AST | The final token of a multi-token entity. |
| U NIT | A single-token entity. |
| O UT | A non-entity token. |

Source : https://towardsdatascience.com/named-entity-recognition-with-nltk-and-spacy-8c4a7d88e7da

IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
Impact Factor: 7.301
Volume 3, Issue 5, April 2023

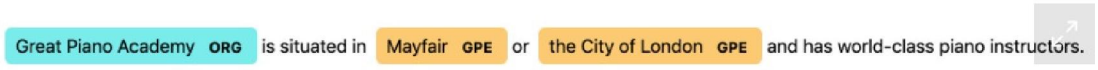
Simply import the spacy and load model, use the nlp to parse the text, then loop over each entity to print its label.

```
Python

>>> piano_class_text = ('Great Piano Academy is situated'
...     ' in Mayfair or the City of London and has'
...     ' world-class piano instructors.')
>>> piano_class_doc = nlp(piano_class_text)
>>> for ent in piano_class_doc.ents:
...     print(ent.text, ent.start_char, ent.end_char,
...           ent.label_, spacy.explain(ent.label_))
...
Great Piano Academy 0 19 ORG Companies, agencies, institutions, etc.
Mayfair 35 42 GPE Countries, cities, states
the City of London 46 64 GPE Countries, cities, states
```

In the preceding illustration, ent is a Span object with following attributes
the text' displays the entity's Unicode text representation.
The character offset for the start of the entity is indicated by the term start_char.
The character offset for the entity's end is indicated by the term "end_char." label_ provides the entity's label.
A descriptive description of an entity label is provided by spacy.explain. The list of entity classes in the spaCy model has already been trained. DisplaCy can be used to represent various entities:

```
Python                                                    >>>

>>> displacy.serve(piano_class_doc, style='ent')
```

If you open `http://127.0.0.1:5000` in your browser, then you can see the visualization:




displaCy: Named Entity Recognition Demo

Due to its speed, usability, accuracy, and adaptability, spaCy is a strong and sophisticated toolkit that is rapidly gaining favour for NLP applications.

## IV. RESULT AND DISCUSSION

4.1 Result:

We have found that when videos are converted to text, the summaries produced are simple enough for readers to understand. Regardless of how brief the summary is, it was producing proper tags for the videos. These outcomes rely on extractive summarizers, thus there is still more room for improvement in our approach when tested with new and updated summarizers, pre-trained models, or comprehensive datasets. The model can be improved using a variety of machine learning techniques, such as fine-tuning. We therefore anticipate that our findings will deepen our understanding and motivate models to perform well.

IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
Impact Factor: 7.301
Volume 3, Issue 5, April 2023

### 4.2 Discussion

Our model could accurately predict the results of a video to text conversion given the present state of computing resources and pre-trained open source code. The findings, however, might change and the new model might outperform the old one if given access to more resources and high-end pre-trained models. Our strategy was to show how video data could be converted into text data, a remarkable finding of our research.

## V. CONCLUSION

The goal of the model is to create an audio file from video and take the input audio and extract the entities. Using NLP techniques, applications must process these videos and generate audio files & text transcripts from the audio file and the related entities. Extract entities from this text, then add pertinent tags to the videos. This information will be used to improve the content recommendations.

Use Cases of Named Entity Recognition:

1. Classifying content for news providers
2. Efficient Search Algorithms
3. Customer Support
4. Powering Content Recommendations
5. Research Papers

**Limitations**: This model only works for Hindi and English language.

**Future scope**: Working on this model to get results from other vernacular languages too.

## REFERENCES

**[1].** wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations
**[2].** https://towardsdatascience.com/named-entity-recognition-with-nlt k-and-spacy-8c4a7d88e7da
**[3].** J. Li, A. Sun, J. Han and C. Li, "A Survey on Deep Learning for Named Entity Recognition, " in IEEE
**[4].** Transactions on Knowledge and Data Engineering, vol. 34, no. 1, pp. 50-70, 1 Jan. 2022, DOI:
**[5].** 10.1109/TKDE.2020.2981314.
**[6].** [ https://medium.com/sciforce/towards-automatic-text-summarizatio n-extractive-methods-e8439cd54715
**[7].** https://www.assemblyai.com/blog/assemblyai-and-python-in-5-minutes/
**[8].** https://www.tensorflow.org/text/tutorials/classify_text_with_bert
**[9].** https://towardsdatascience.com/understanding-encoder-decoder-sequence-to-sequence-model-679e04af4346
**[10].** https://monkeylearn.com/blog/entity-extraction
**[11].** https://towardsdatascience.com/named-entity-recognition-with-nltk-and-spacy-8c4a7d88e7da