Лабораторна робота № 1. Дослідження платформи Hugging Face[1]

Мета: ознайомитися з платформою Hugging Face, дослідити основні розділи (Models, Datasets, Spaces) та навчитися використовувати ресурси платформи для вирішення задач машинного навчання.

Завдання для всіх варіантів

1. ***Дослідження розділу Models.***

- Вивчіть різні категорії моделей (наприклад, Text Classification, Named Entity Recognition, Question Answering тощо).
- Оберіть дві різні моделі з різних категорій.
- Для кожної моделі:
    - Опишіть її призначення та архітектуру.
    - Знайдіть приклад використання моделі та спробуйте його відтворити.
    - Проаналізуйте переваги та обмеження моделі.

2. ***Дослідження розділу Datasets.***

- Ознайомтесь з різними типами наборів даних (текстові, аудіо, зображення тощо).
- Оберіть два набори даних різних типів.
- Для кожного набору даних:
    - Опишіть його структуру та призначення.
    - Завантажте зразок даних та проведіть базовий аналіз (наприклад, розмір, розподіл класів, типи ознак).
    - Запропонуйте потенційне застосування цього набору даних для вирішення реальних задач.

[1] Hugging Face – платформа для спільної праці над моделями, наборами даних і додатками. URL: https://huggingface.co/

3. ***Дослідження розділу Spaces.***

- Перегляньте різні проекти в Spaces.
- Оберіть два цікавих проекти.
- Для кожного проекту:
    - Опишіть його функціональність та технології, що використовуються.
    - Спробуйте взаємодіяти з проектом та проаналізуйте його роботу.

Хід роботи

1. ***Дослідження розділу Models***

   - Для дослідження було обрано такі моделі:

     BERT (bert-base-uncased) для класифікації тексту.

     YOLOv5 для виявлення об’єктів на зображеннях.

   - Дослідження моделі BERT:

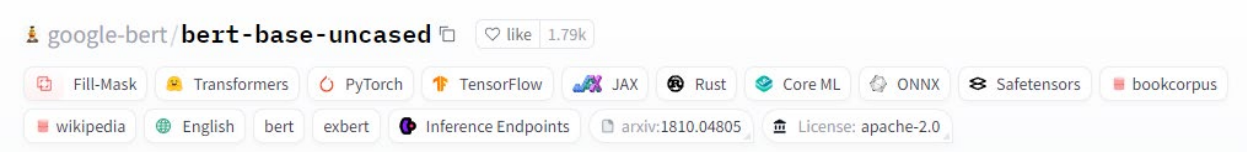

Рисунок 1.1. Модель BERT (bert-base-uncased) на платформі Hugging Face

Опис моделі: BERT (Bidirectional Encoder Representations from Transformers) – це модель для обробки природної мови, яка використовує двонаправлене навчання на основі трансформерів.

Приклад використання:

```
from transformers import AutoTokenizer, AutoModelForSequenceClassification
import torch

# Завантаження моделі та токенізатора
tokenizer = AutoTokenizer.from_pretrained("bert-base-uncased")
model = AutoModelForSequenceClassification.from_pretrained("bert-base-
uncased")

# Підготовка вхідних даних
text = "I love this movie!"
inputs = tokenizer(text, return_tensors="pt")

# Отримання передбачення
outputs = model(**inputs)
predictions = torch.nn.functional.softmax(outputs.logits, dim=-1)
print(predictions)
```

```
tensor([[0.3623, 0.6377]], grad_fn=<SoftmaxBackward0>)
```

- Аналіз:

  Переваги:

  1. BERT досягає state-of-the-art результатів у багатьох завданнях NLP.
  2. BERT враховує контекст слова з обох боків, що дозволяє краще розуміти значення слів у контексті.
  3. Попередньо навчену модель можна легко адаптувати до специфічних завдань з меншою кількістю даних.
  4. Існують версії BERT, навчені на багатьох мовах, що дозволяє використовувати модель для різних мов.
  5. BERT може бути використаний для широкого спектру завдань NLP, включаючи класифікацію, Named Entity Recognition, Question Answering тощо.

  Обмеження:

  1. BERT вимагає значних обчислювальних ресурсів для навчання та інференсу.
  2. Для ефективної роботи BERT часто потребує GPU.
  3. Стандартний BERT обмежений 512 токенами, що може бути проблемою для довгих текстів.
  4. Як і багато інших моделей глибокого навчання, BERT може бути "чорною скринькою", ускладнюючи розуміння його рішень.
  5. BERT може успадковувати упередження з навчальних даних, що потребує уваги при використанні в чутливих областях.

- Дослідження моделі YOLOv5:

Рисунок 1.2. Модель YOLOv5 на платформі Hugging Face

Опис моделі: YOLOv5 (You Only Look Once version 5) – це сучасна модель для виявлення об’єктів у реальному часі. Вона є однією з найшвидших та найточніших моделей для цього завдання.

Приклад використання:

```
import torch
import cv2
import numpy as np
import requests

# Завантаження моделі
model = torch.hub.load('ultralytics/yolov5', 'yolov5s')

# Завантаження зображення з URL за допомогою OpenCV
url = 'https://ultralytics.com/images/zidane.jpg'
response = requests.get(url)
img_arr = np.asarray(bytearray(response.content), dtype=np.uint8)
img = cv2.imdecode(img_arr, cv2.IMREAD_COLOR)

# Перетворення зображення в RGB (якщо це потрібно для моделі)
img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)

# Виконання виявлення
results = model(img)

# Відображення результатів
results.print()  # Виводить інформацію про знайдені об'єкти
results.show()  # Відображає зображення з позначеними об'єктами

# Отримання результатів у вигляді pandas DataFrame
df = results.pandas().xyxy[0]
print(df)
```

```
image 1/1: 720x1280 2 persons, 1 tie, 1 cell phone
Speed: 3.4ms pre-process, 504.7ms inference, 1.7ms NMS per image at shape
(1, 3, 384, 640)
```

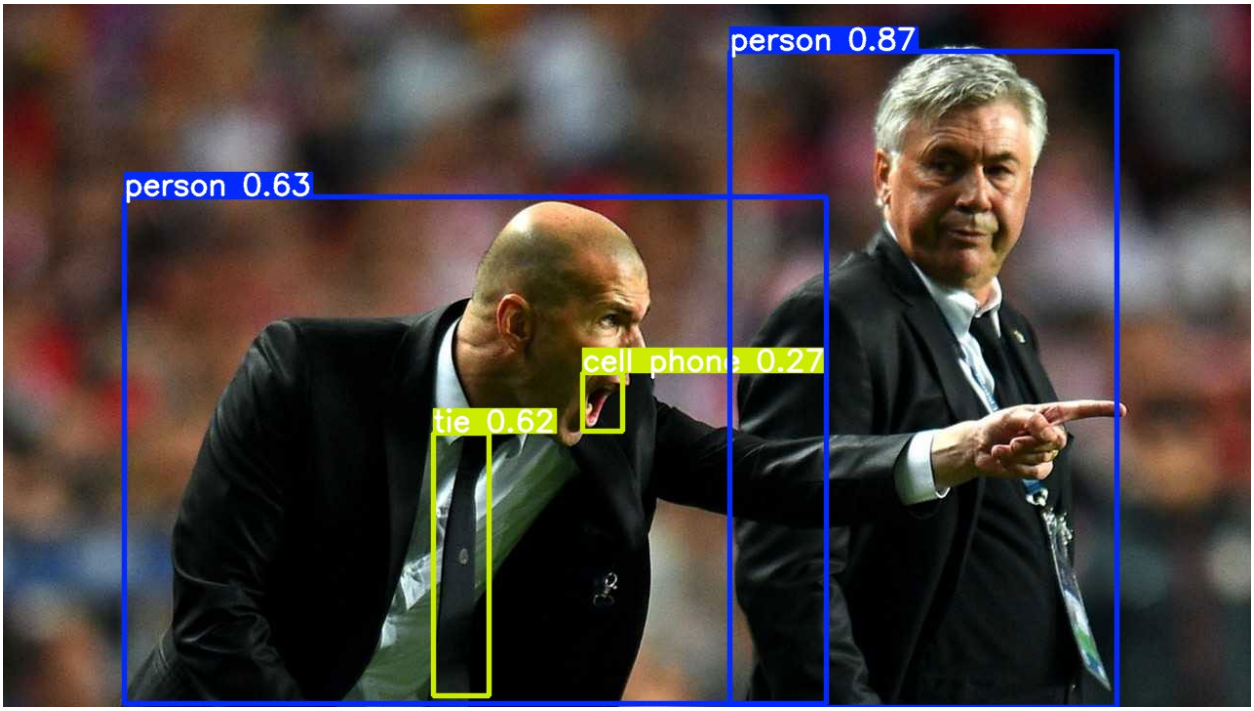

Рисунок 1.2. Результат роботи моделі YOLOv5

- Аналіз:

  Переваги:

  1. YOLOv5 здатна працювати в режимі реального часу навіть на відносно слабких пристроях.
  2. Модель досягає високої точності виявлення об'єктів.
  3. Доступні різні версії моделі (від найлегшої до найважчої) для різних сценаріїв використання.
  4. Модель легко інтегрувати в різні проекти завдяки простому API.
  5. Постійні оновлення та покращення від спільноти розробників.

  Обмеження:

  1. У порівнянні з деякими іншими моделями, YOLOv5 може гірше виявляти дуже маленькі об'єкти.
  2. Для досягнення найкращих результатів потрібен великий набір розмічених даних.
  3. У складних сценах модель може іноді виявляти неіснуючі об'єкти.
  4. Для виявлення специфічних об'єктів може знадобитися додаткове навчання.

2. ***Дослідження розділу Datasets***

- Для дослідження були обрані наступні набори даних:

  IMDB Movie Reviews для аналізу настроїв.

  COCO (Common Objects in Context) для комп'ютерного зору.

- Дослідження набору даних IMDB Movie Reviews:

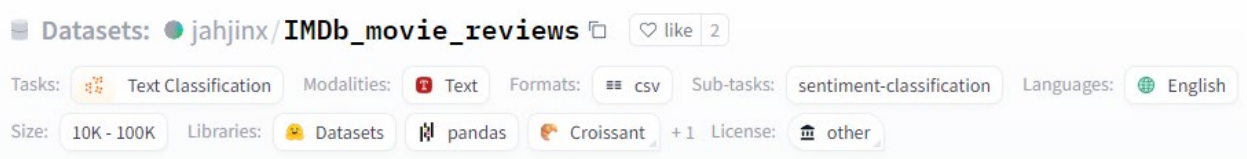

Рисунок 1.3. Набір даних IMDb movie reviews на платформі Hugging Face

Опис набору даних: IMDB Movie Reviews – це набір даних, що містить 50,000 відгуків про фільми з бінарними мітками настрою (позитивний/негативний).

Приклад використання:

```
from datasets import load_dataset

# Завантаження набору даних
dataset = load_dataset("imdb")

# Аналіз розміру
print(f"Розмір тренувального набору: {len(dataset['train'])}")
print(f"Розмір тестового набору: {len(dataset['test'])}")

# Аналіз розподілу класів
positive = sum(1 for x in dataset['train'] if x['label'] == 1)
negative = sum(1 for x in dataset['train'] if x['label'] == 0)
print(f"Позитивні відгуки: {positive}")
print(f"Негативні відгуки: {negative}")

# Приклад даних
print(dataset['train'][0])
```

```
Розмір тренувального набору: 25000
Розмір тестового набору: 25000
Позитивні відгуки: 12500
Негативні відгуки: 12500
```

- Потенційне застосування: цей набір даних може бути використаний для навчання моделей аналізу настроїв, які можуть застосовуватися в системах рекомендацій, моніторингу соціальних медіа тощо.

- Дослідження набору даних COCO (Common Objects in Context):

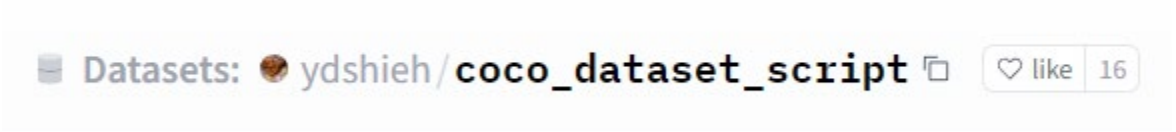

Рисунок 1.4. Набір даних COCO (Common Objects in Context) на платформі Hugging Face

Опис набору даних: COCO (Common Objects in Context) – це великомасштабний набір даних для виявлення об'єктів, сегментації та підписування зображень. Він містить понад 330 тисяч зображень, з яких понад 200 тисяч розмічені.

Приклад використання:

```
import requests
import zipfile
import os
from pycocotools.coco import COCO
import numpy as np
import skimage.io as io
import matplotlib.pyplot as plt

# Створення папок для даних
!mkdir -p coco_dataset
!mkdir -p coco_dataset/annotations
!mkdir -p coco_dataset/images

# Завантаження і розпакування набору даних COCO (анотації та зображення)
coco_url =
'http://images.cocodataset.org/annotations/annotations_trainval2017.zip'
image_url = 'http://images.cocodataset.org/zips/val2017.zip'

# Завантаження та розпаковка анотацій
ann_path = 'coco_dataset/annotations/instances_val2017.json'
if not os.path.exists(ann_path):
    r = requests.get(coco_url)
    with open('coco_dataset/annotations.zip', 'wb') as f:
        f.write(r.content)
    with zipfile.ZipFile('coco_dataset/annotations.zip', 'r') as zip_ref:
```

```
        zip_ref.extractall('coco_dataset/annotations')

# Перевірка, чи файл анотацій був успішно розпакований
if os.path.exists(ann_path):
    print("Файл анотацій успішно знайдено!")
else:
    print("Файл анотацій не знайдено. Перевірте шлях до файлу.")

# Завантаження та розпаковка зображень
if not os.path.exists('coco_dataset/images/val2017'):
    r = requests.get(image_url)
    with open('coco_dataset/images.zip', 'wb') as f:
        f.write(r.content)
    with zipfile.ZipFile('coco_dataset/images.zip', 'r') as zip_ref:
        zip_ref.extractall('coco_dataset/images')

# Перевірка, чи зображення були успішно розпаковані
if os.path.exists('coco_dataset/images/val2017'):
    print("Зображення успішно завантажені та розпаковані!")
else:
    print("Помилка завантаження зображень. Перевірте шлях до файлів.")

# Ініціалізація COCO API
dataDir = 'coco_dataset'
dataType = 'val2017'
annFile =
'/content/coco_dataset/annotations/annotations/instances_val2017.json'
coco = COCO(annFile)

# Отримання інформації про категорії
cats = coco.loadCats(coco.getCatIds())
cat_names = [cat['name'] for cat in cats]
print(f"COCO категорії: {', '.join(cat_names)}")

# Аналіз кількості анотацій для кожної категорії
ann_counts = [len(coco.getAnnIds(catIds=coco.getCatIds(catNms=[cat]))) for
cat in cat_names]
for cat, count in zip(cat_names, ann_counts):
    print(f"{cat}: {count} анотацій")

# Візуалізація випадкового зображення з анотаціями
catIds = coco.getCatIds(catNms=['person', 'dog', 'skateboard'])
imgIds = coco.getImgIds(catIds=catIds)
img = coco.loadImgs(imgIds[np.random.randint(0, len(imgIds))])[0]

I = io.imread(f"{dataDir}/images/{dataType}/{img['file_name']}")
plt.imshow(I); plt.axis('off')
annIds = coco.getAnnIds(imgIds=img['id'], catIds=catIds, iscrowd=None)
anns = coco.loadAnns(annIds)
```

```
coco.showAnns(anns)
plt.show()
```

```
COCO категорії: person, bicycle, car, motorcycle, airplane, bus, train,
truck, boat, traffic light, fire hydrant, stop sign, parking meter, bench,
bird, cat, dog, horse, sheep, cow, elephant, bear, zebra, giraffe,
backpack, umbrella, handbag, tie, suitcase, frisbee, skis, snowboard,
sports ball, kite, baseball bat, baseball glove, skateboard, surfboard,
tennis racket, bottle, wine glass, cup, fork, knife, spoon, bowl, banana,
apple, sandwich, orange, broccoli, carrot, hot dog, pizza, donut, cake,
chair, couch, potted plant, bed, dining table, toilet, tv, laptop, mouse,
remote, keyboard, cell phone, microwave, oven, toaster, sink,
refrigerator, book, clock, vase, scissors, teddy bear, hair drier,
toothbrush
person: 11004 анотацій
bicycle: 316 анотацій
car: 1932 анотацій
motorcycle: 371 анотацій
airplane: 143 анотацій
bus: 285 анотацій
train: 190 анотацій
…
```

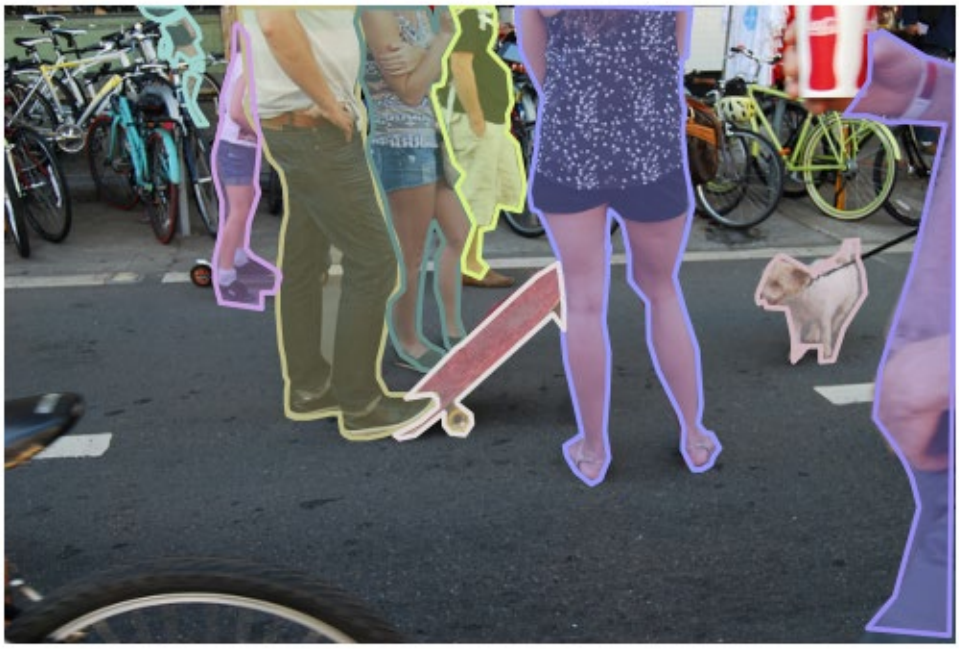

Рисунок 1.5. Приклад зображення з набору даних COCO, що було проанотовано

- Потенційне застосування: 1) Навчання та оцінка моделей виявлення об'єктів та сегментації зображень; 2) Розробка систем комп'ютерного зору для розпізнавання об'єктів у реальному світі; 3) Дослідження в області розуміння сцен та контексту зображень; 4) Створення систем автоматичного

підписування зображень; 5) Розробка алгоритмів для автономних транспортних засобів.

3. *Дослідження розділу Spaces*

- Для дослідження були обрані наступні простори:

Automatic Speech Recognition для автоматичного розпізнавання мовлення.

Latent Navigation для генерації послідовностей зображень за допомогою Latent Directions.

- Дослідження простору Automatic Speech Recognition:

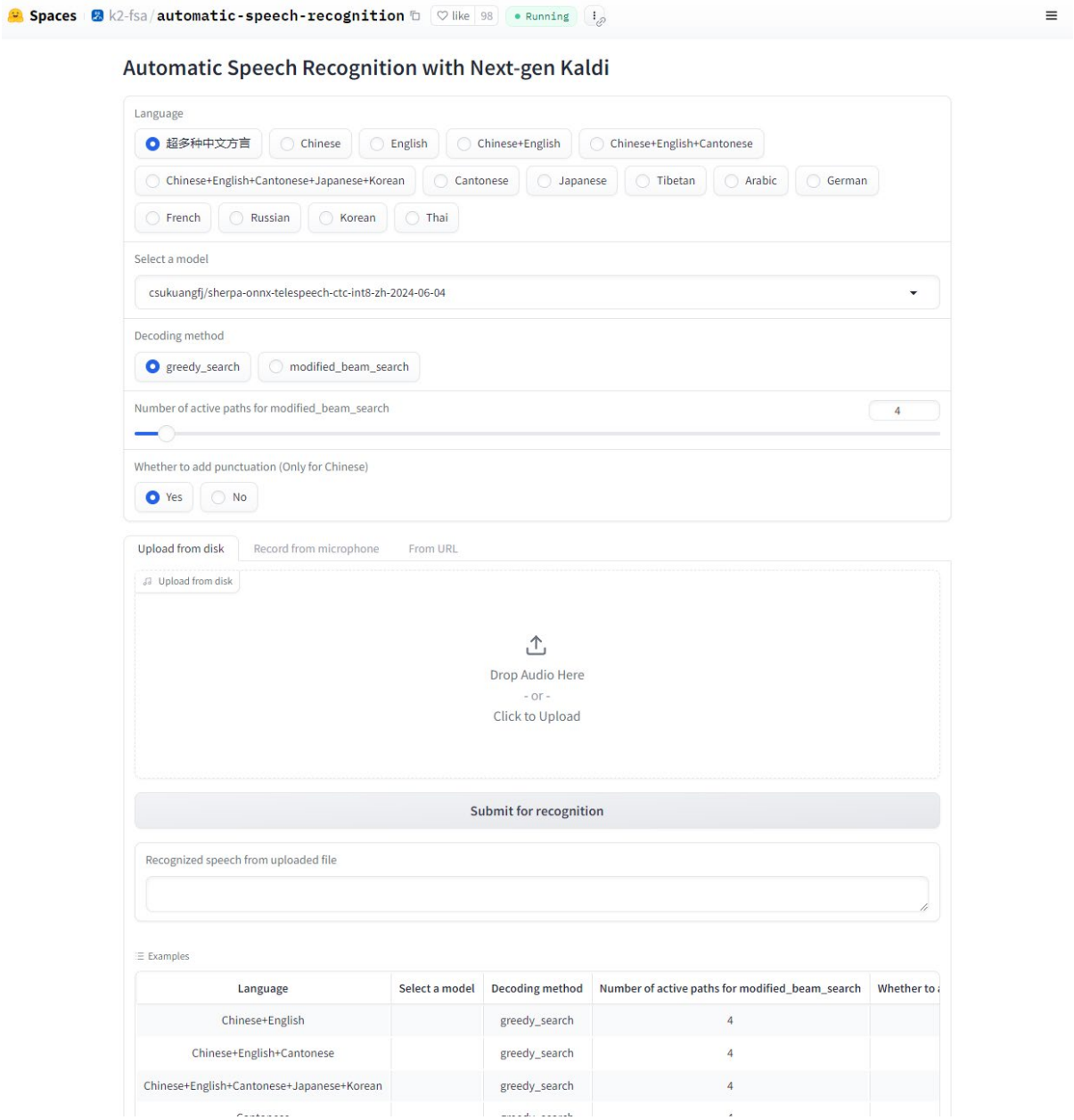

Рисунок 1.6. Простір Automatic Speech Recognition на платформі Hugging Face

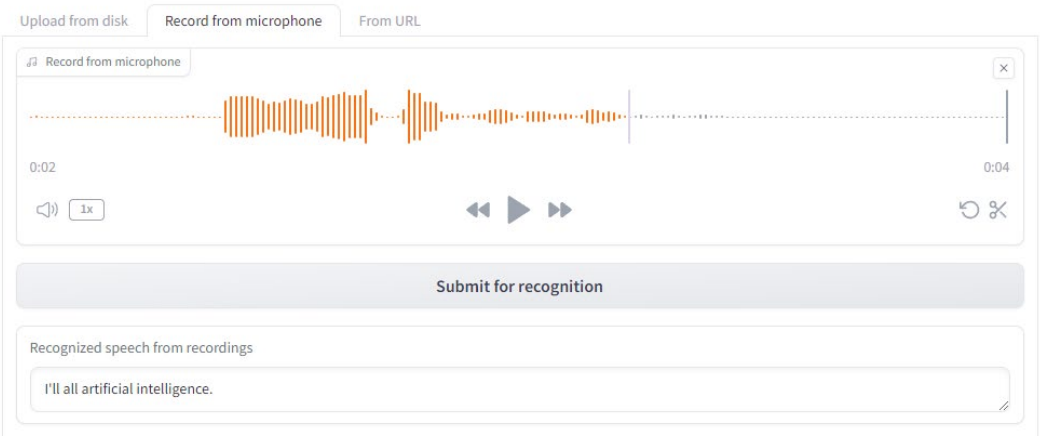

Рисунок 1.7. Приклад розпізнавання тексту англійською мовою

- Опис простору: цей проект використовує комплекс різноманітних моделей, зокрема Whisper від OpenAI для транскрибації аудіо в текст. Моделі здатні розпізнавати різні мови та діалекти.

- Дослідження простору Latent Navigation:

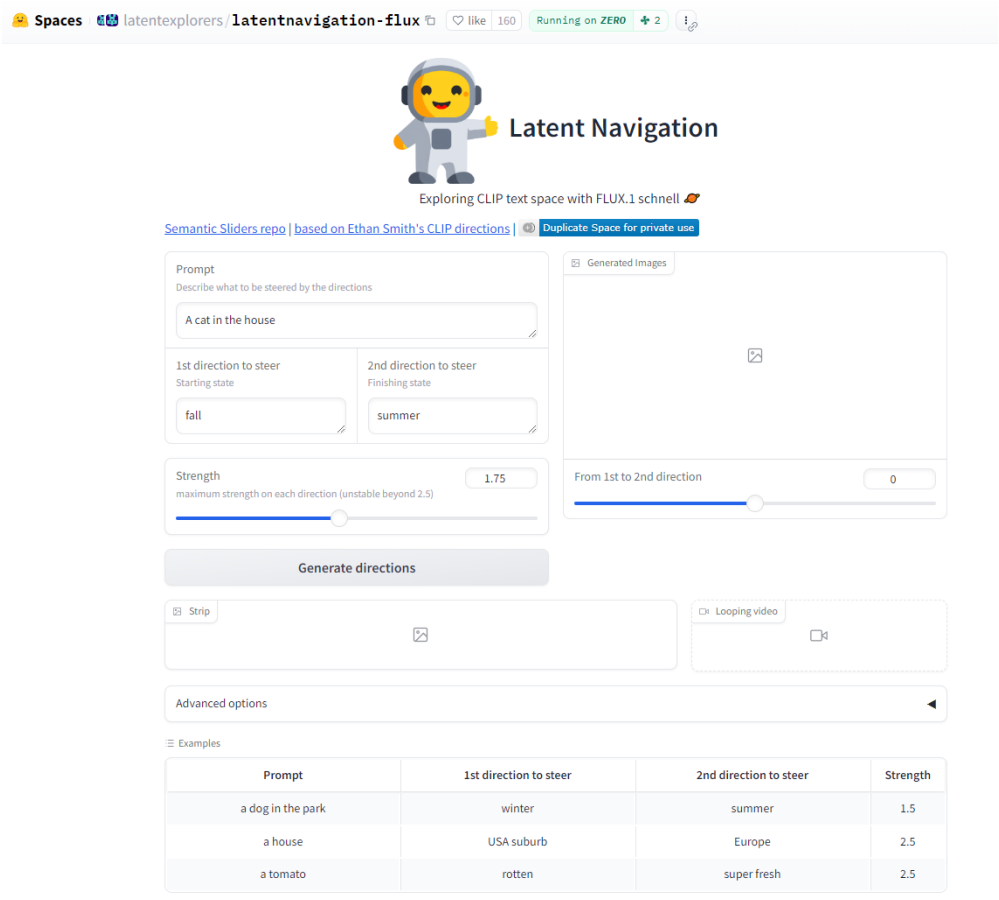

Рисунок 1.8. Простір Automatic Speech Recognition на платформі Hugging Face

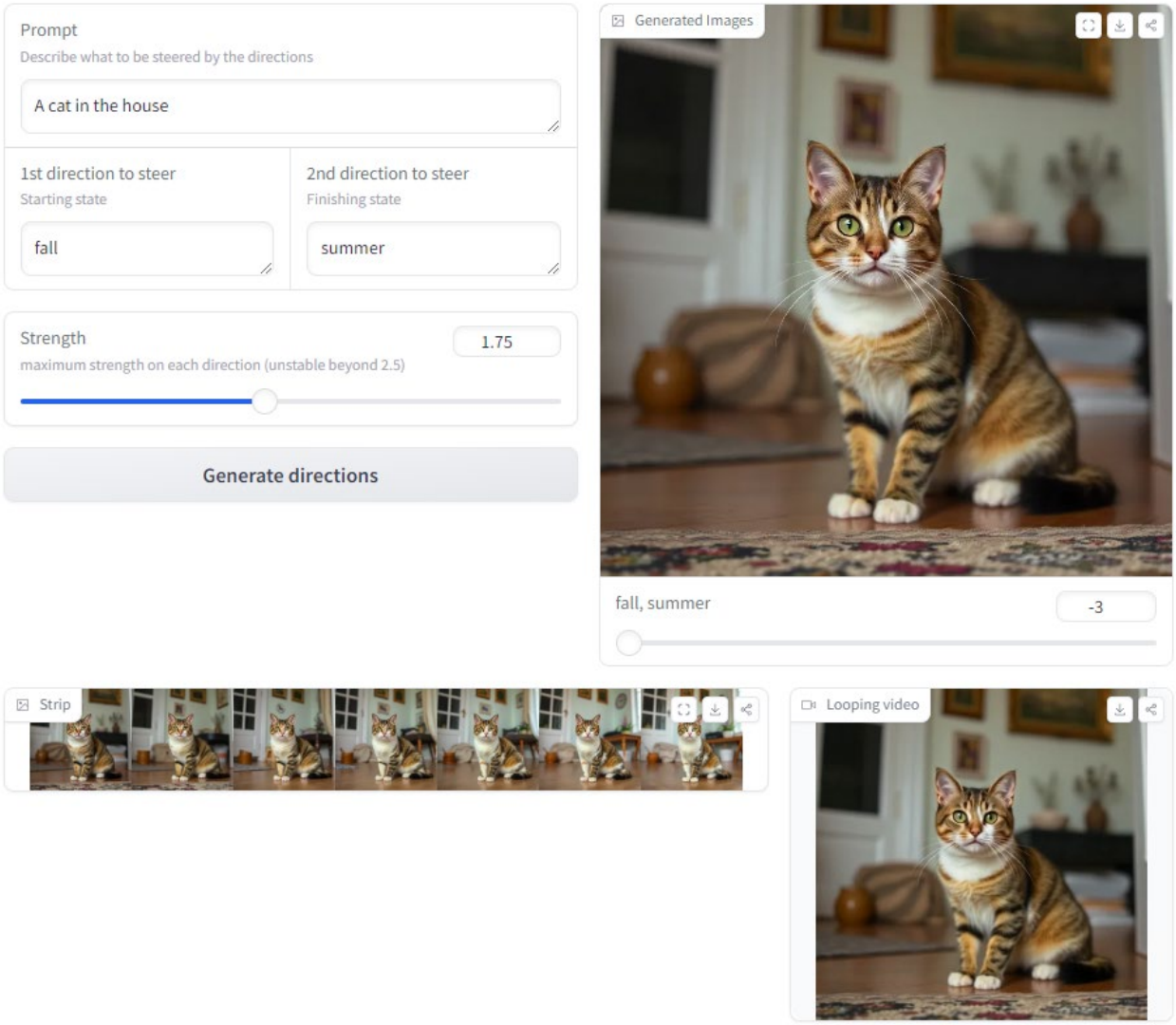

Рисунок 1.9. Приклад створення послідовності зображення кота в будинку у різні пори року

- Опис простору: цей проект використовує переміщення через простір CLIP, PCA та латентні напрямки для створення послідовності зображень в динаміці за одним промптом.

Висновки: …