Transformers: তারা কী কী করতে পারে?

এই সেকশনে আমরা দেখব Transformer মডেলগুলো আসলে কী কী কাজ করতে পারে এবং Transformers লাইব্রেরির প্রথম ও সবচেয়ে সহজ টুল `pipeline()` ফাংশন ব্যবহার করব।

Transformers সর্বত্র

Transformer মডেলগুলো এখন অনেক ধরনের কাজ (multi-modal) এ ব্যবহার হচ্ছে, যেমন:

- Natural Language Processing (NLP)
- Computer Vision (ইমেজ প্রসেসিং)
- Audio Processing (স্পিচ/অডিও)
- এমনকি Text + Image একসাথে (multimodal)

অনেক কোম্পানি এবং সংস্থা Hugging Face এবং Transformer মডেল ব্যবহার করে। তারা কমিউনিটিতে অবদান রাখে—নিজেদের ট্রেন করা মডেল Hub–এ শেয়ার করে।

Model Hub কী?

Transformers লাইব্রেরি আপনাকে Hub–এ থাকা শেয়ার করা মডেলগুলো ব্যবহার করতে এবং চাইলে নিজের মডেল আপলোড করতে সাহায্য করে। Model Hub–এ লক্ষ লক্ষ pretrained মডেল আছে, যে কেউ ডাউনলোড করে ব্যবহার করতে পারে।

গুরুত্বপূর্ণ সতর্কতা:

- Hugging Face Hub শুধু Transformer–এর জন্য সীমাবদ্ধ না
- যে কেউ যেকোন ধরনের মডেল বা ডেটাসেট শেয়ার করতে পারে
- huggingface.co অ্যাকাউন্ট খুললে Hub–এর সব ফিচার ব্যবহার করতে পারবেন

Pipeline দিয়ে কাজ করা

`pipeline()` কী এবং কেন গুরুত্বপূর্ণ?

Transformers লাইব্রেরির সবচেয়ে বেসিক অবজেক্ট হলো `pipeline()`। এটি একটি মডেলকে তার প্রয়োজনীয় preprocessing এবং postprocessing এর সাথে জুড়ে দেয়, ফলে আমরা সরাসরি টেক্সট ইনপুট দিতে পারি এবং বোঝার মতো আউটপুট পেতে পারি।

উদাহরণ: **Sentiment Analysis**

```
from transformers import pipeline

classifier = pipeline("sentiment-analysis")
classifier("I've been waiting for a HuggingFace course my whole life.")
```

আউটপুট:

```
[{'label': 'POSITIVE', 'score': 0.9598047137260437}]
```

একসাথে অনেকগুলো বাক্যও পাঠানো যায় **(Batch)**

```
classifier(
    ["I've been waiting for a HuggingFace course my whole life.", "I hate this so much!"]
)
```

আউটপুট:

```
[{'label': 'POSITIVE', 'score': 0.9598047137260437},
 {'label': 'NEGATIVE', 'score': 0.9994558095932007}]
```

ডিফল্ট মডেল কীভাবে সিলেক্ট হয়**?**

ডিফল্টভাবে এই pipeline এমন একটি pretrained মডেল বেছে নেয়, যেটা English sentiment analysis এর জন্য আগে থেকেই fine-tuned।

আরেকটা গুরুত্বপূর্ণ ব্যাপার:

- আপনি যখন `classifier` অবজেক্ট বানান, তখন মডেল ডাউনলোড হয় এবং cache হয়ে যায়
- পরে আবার কোড চালালে cache থেকে ব্যবহার হবে, আবার ডাউনলোড লাগবে না

Pipeline কীভাবে কাজ করে**? (**৩টি মূল ধাপ**)**

Pipeline–এ কোনো টেক্সট পাঠালে সাধারণত এই তিনটা ধাপ ঘটে:

1. টেক্সটকে এমন ফরম্যাটে প্রি-প্রসেস করা হয়, যেটা মডেল বুঝতে পারে
2. সেই ইনপুট মডেলে পাঠানো হয়
3. মডেলের prediction কে পোস্ট-প্রসেস করে মানুষের বোঝার মতো আউটপুট বানানো হয়

বিভিন্ন **modality–**র জন্য **pipeline**

`pipeline()` শুধু টেক্সট নয়—ইমেজ, অডিও, এমনকি multimodal টাস্কও সাপোর্ট করে। এই কোর্সে আমরা মূলত টেক্সট নিয়ে কাজ করব, কিন্তু Transformer আর্কিটেকচারের ক্ষমতা বুঝতে অন্য modality–গুলোর আইডিয়াও জানা দরকার।

নিচে কী কী pipeline পাওয়া যায় তার একটা ধারণা দেওয়া হলো। (সম্পূর্ণ ও আপডেটেড লিস্ট ডকুমেন্টেশনে থাকে।)

Text pipelines

- `text-generation`: prompt থেকে টেক্সট তৈরি
- `text-classification`: টেক্সটকে পূর্বনির্ধারিত ক্যাটাগরিতে ভাগ করা
- `summarization`: বড় লেখা ছোট করে সারাংশ বানানো
- `translation`: এক ভাষা থেকে আরেক ভাষায় অনুবাদ
- `zero-shot-classification`: আগে থেকে নির্দিষ্ট লেবেলে ট্রেন না করেও লেবেল দিয়ে ক্লাসিফাই
- `feature-extraction`: টেক্সটের vector representation বের করা

Image pipelines

- `image-to-text`: ছবির বর্ণনা তৈরি
- `image-classification`: ছবিতে কী আছে শনাক্ত করা
- `object-detection`: অবজেক্ট কোথায় আছে + কী অবজেক্ট সেটাও শনাক্ত করা

Audio pipelines

- `automatic-speech-recognition`: কথা থেকে লেখা (speech to text)
- `audio-classification`: অডিওকে ক্যাটাগরিতে ভাগ করা
- `text-to-speech`: লেখা থেকে কথা তৈরি

Multimodal pipelines

- `image-text-to-text`: ছবি + টেক্সট প্রম্পট নিয়ে রেসপন্স তৈরি

Zero-shot classification (লেবেলড ডেটা না থাকলেও ক্লাসিফাই)

এখন আমরা একটি একটু কঠিন কিন্তু বাস্তব জীবনে খুব কমন সমস্যা দেখব: লেবেলড ডেটা নেই।

বাস্তবে টেক্সট লেবেলিং করতে সময় লাগে এবং ডোমেইন এক্সপার্টও লাগে। এই ক্ষেত্রে `zero-shot-classification` pipeline খুব শক্তিশালী—কারণ আপনি নিজেই লেবেল তালিকা দিয়ে দিতে পারেন; pretrained মডেলের ডিফল্ট লেবেলের উপর নির্ভর করতে হয় না।

```
from transformers import pipeline

classifier = pipeline("zero-shot-classification")
classifier(
    "This is a course about the Transformers library",
    candidate_labels=["education", "politics", "business"],
)
```

আউটপুট:

```
{'sequence': 'This is a course about the Transformers library',
 'labels': ['education', 'business', 'politics'],
 'scores': [0.8445963859558105, 0.111976258456707, 0.043427448719739914]}
```

এটাকে “zero-shot” বলা হয় কারণ:

- আপনাকে নিজের ডেটায় মডেল fine-tune করতে হচ্ছে না
- আপনি যে কোনো লেবেল লিস্ট দিলে মডেল সেগুলোর জন্য score বের করে দিতে পারে

Text generation (প্রম্পট থেকে লেখা তৈরি)

Text generation এ আপনি একটি প্রম্পট দেন, মডেল বাকিটা অটো-কমপ্লিট করে। এটা অনেকটা ফোনের predictive text এর মতো, তবে অনেক বেশি শক্তিশালী।

এখানে randomness থাকে, তাই সবসময় একই আউটপুট নাও আসতে পারে।

```
from transformers import pipeline

generator = pipeline("text-generation")
```

generator("In this course, we will teach you how to")

আউটপুট (একটা উদাহরণ):

```
[{'generated_text': 'In this course, we will teach you how to understand and use '
                    'data flow and data interchange when handling user data. We '
                    'will be working with one or more of the most commonly used '
                    'data flows — data flows of various types, as seen by the '
                    'HTTP'}]
```

আপনি নিয়ন্ত্রণ করতে পারেন:

- `num_return_sequences`: কয়টা আলাদা আউটপুট চান
- `max_length`: আউটপুট মোট কত লম্বা হবে

Hub থেকে যেকোনো মডেল pipeline–এ ব্যবহার করা

আগের উদাহরণগুলোতে ডিফল্ট মডেল ব্যবহার হয়েছে। কিন্তু আপনি চাইলে Hub থেকে নির্দিষ্ট মডেল সিলেক্ট করে pipeline–এ বসাতে পারেন।

উদাহরণ হিসেবে `HuggingFaceTB/SmolLM2-360M` মডেল:

```
from transformers import pipeline

generator = pipeline("text-generation", model="HuggingFaceTB/SmolLM2-360M")
generator(
    "In this course, we will teach you how to",
    max_length=30,
    num_return_sequences=2,
)
```

আউটপুট:

```
[{'generated_text': 'In this course, we will teach you how to manipulate the world and '
                    'move your mental and physical capabilities to your advantage.'},
 {'generated_text': 'In this course, we will teach you how to become an expert and '
                    'practice realtime, and with a hands on experience on both real '
                    'time and real'}]
```

আপনি চাইলে Hub–এ language tag ফিল্টার দিয়ে অন্য ভাষার টেক্সট জেনারেশন মডেলও খুঁজে নিতে পারেন। অনেক multilingual মডেল আছে যেগুলো একাধিক ভাষা সাপোর্ট করে।

Inference Providers (ব্রাউজার থেকে মডেল টেস্ট)

Hugging Face ওয়েবসাইটে Inference Providers ব্যবহার করে আপনি ব্রাউজারেই মডেল টেস্ট করতে পারবেন—নিজের টেক্সট দিয়ে দেখে নিতে পারবেন মডেল কীভাবে প্রসেস করছে।

Widget–এর পেছনে যে ইনফারেন্স সার্ভিস কাজ করে, সেটি paid পণ্য হিসেবেও পাওয়া যায় (প্রোডাকশন workflow–এ লাগতে পারে)।

Mask filling (fill-mask)

`fill-mask` pipeline দিয়ে বাক্যে ফাঁকা জায়গা পূরণ করা হয়—মানে `<mask>` টোকেনের জায়গায় মডেল সম্ভাব্য শব্দ প্রেডিক্ট করে।

```
from transformers import pipeline

unmasker = pipeline("fill-mask")
unmasker("This course will teach you all about <mask> models.", top_k=2)
```

আউটপুট:

```
[{'sequence': 'This course will teach you all about mathematical models.',
  'score': 0.19619831442832947,
  'token': 30412,
  'token_str': ' mathematical'},
 {'sequence': 'This course will teach you all about computational models.',
  'score': 0.04052725434303284,
  'token': 38163,
  'token_str': ' computational'}]
```

এখানে `top_k` মানে:

- আপনি কতগুলো সম্ভাব্য শব্দ দেখাতে চান

গুরুত্বপূর্ণ নোট:

- বিভিন্ন মডেলে mask টোকেন ভিন্ন হতে পারে
- তাই নতুন মডেল ব্যবহার করলে widget থেকে mask word যাচাই করে নেওয়া ভালো

Named Entity Recognition (NER)

NER এর কাজ হলো বাক্যের ভেতর থেকে entity বের করা—যেমন ব্যক্তি, সংস্থা, জায়গা।

```
from transformers import pipeline

ner = pipeline("ner", grouped_entities=True)
ner("My name is Sylvain and I work at Hugging Face in Brooklyn.")
```

আউটপুট:

```
[{'entity_group': 'PER', 'score': 0.99816, 'word': 'Sylvain', 'start': 11, 'end': 18},
 {'entity_group': 'ORG', 'score': 0.97960, 'word': 'Hugging Face', 'start': 33, 'end': 45},
 {'entity_group': 'LOC', 'score': 0.99321, 'word': 'Brooklyn', 'start': 49, 'end': 57}
]
```

এখানে:

- Sylvain = PER (Person)
- Hugging Face = ORG (Organization)
- Brooklyn = LOC (Location)

`grouped_entities=True` কেন?
কারণ অনেক entity একাধিক শব্দের হয় (Hugging + Face)
আর tokenizer কখনো শব্দকে subword এ ভেঙে ফেলে (যেমন Sylvain → S, ##yl, ##va, ##in)
Post-processing এ pipeline এগুলো আবার একসাথে গ্রুপ করে সুন্দর entity বানায়।

Question Answering (context থেকে উত্তর বের করা)

এই pipeline একটি প্রশ্নকে দেওয়া context থেকে উত্তর “extract” করে।

```
from transformers import pipeline

question_answerer = pipeline("question-answering")
question_answerer(
    question="Where do I work?",
    context="My name is Sylvain and I work at Hugging Face in Brooklyn",
)
```

আউটপুট:

```
{'score': 0.6385916471481323, 'start': 33, 'end': 45, 'answer': 'Hugging Face'}
```

গুরুত্বপূর্ণ নোট:

- এটি context থেকে তথ্য “বের করে” আনে
- নতুন করে উত্তর বানায় না (generative নয়)

Summarization (সারসংক্ষেপ)

Summarization মানে বড় লেখাকে ছোট করা—কিন্তু মূল তথ্য ধরে রাখা।

```
from transformers import pipeline

summarizer = pipeline("summarization")
summarizer(
    """
    America has changed dramatically during recent years. Not only has the number of
    graduates in traditional engineering disciplines such as mechanical, civil,
    electrical, chemical, and aeronautical engineering declined, but in most of
    the premier American universities engineering curricula now concentrate on
    and encourage largely the study of engineering science. As a result, there
    are declining offerings in engineering subjects dealing with infrastructure,
    the environment, and related issues, and greater concentration on high
    technology subjects, largely supporting increasingly complex scientific
    developments. While the latter is important, it should not be at the expense
    of more traditional engineering.

    Rapidly developing economies such as China and India, as well as other
```

```
    industrial countries in Europe and Asia, continue to encourage and advance
    the teaching of engineering. Both China and India, respectively, graduate
    six and eight times as many traditional engineers as does the United States.
    Other industrial countries at minimum maintain their output, while America
    suffers an increasingly serious decline in the number of engineering graduates
    and a lack of well-educated engineers.
"""
)
```

আউটপুট:

```
[{'summary_text': ' America has changed dramatically during recent years . The '
                  'number of engineering graduates in the U.S. has declined in '
                  'traditional engineering disciplines such as mechanical, civil '
                  ', electrical, chemical, and aeronautical engineering . Rapidly '
                  'developing economies such as China and India, as well as other '
                  'industrial countries in Europe and Asia, continue to encourage '
                  'and advance engineering .'}]
```

এখানেও `max_length` বা `min_length` দিয়ে সারসংক্ষেপের দৈর্ঘ্য নিয়ন্ত্রণ করা যায়।

Translation (অনুবাদ)

Translation এর জন্য আপনি language pair দিয়ে ডিফল্ট মডেল নিতে পারেন, কিন্তু সাধারণত Hub থেকে নির্দিষ্ট মডেল বেছে নেওয়াই সবচেয়ে সহজ।

এখানে French → English উদাহরণ:

```
from transformers import pipeline

translator = pipeline("translation", model="Helsinki-NLP/opus-mt-fr-en")
translator("Ce cours est produit par Hugging Face.")
```

আউটপুট:

```
[{'translation_text': 'This course is produced by Hugging Face.'}]
```

Image এবং Audio pipeline উদাহরণ

Transformers শুধু টেক্সট না—ইমেজ ও অডিওতেও কাজ করে।

Image classification

```
from transformers import pipeline

image_classifier = pipeline(
    task="image-classification", model="google/vit-base-patch16-224"
)
result = image_classifier(

"https://huggingface.co/datasets/huggingface/documentation-images/resolve/main/pipeline-cat-chonk.jpeg"
)
print(result)
```

আউটপুট:

```
[{'label': 'lynx, catamount', 'score': 0.43350091576576233},
 {'label': 'cougar, puma, catamount, mountain lion, painter, panther, Felis concolor',
  'score': 0.034796204417943954},
 {'label': 'snow leopard, ounce, Panthera uncia',
  'score': 0.03240183740854263},
 {'label': 'Egyptian cat', 'score': 0.02394474856555462},
 {'label': 'tiger cat', 'score': 0.02288915030658245}]
```

Automatic speech recognition (speech → text)

```
from transformers import pipeline

transcriber = pipeline(
    task="automatic-speech-recognition", model="openai/whisper-large-v3"
)
result = transcriber(
    "https://huggingface.co/datasets/Narsil/asr_dummy/resolve/main/mlk.flac"
)
print(result)
```

আউটপুট:

{'text': ' I have a dream that one day this nation will rise up and live out the true meaning of its creed.'}