`tokenizer` ফাংশন ব্যবহার করে টোকেনাইজেশন, প্যাডিং, এবং ট্রাঙ্কেশন-এর মতো সব কাজ একবারে করা যায়। নিচে কোডসহ বিস্তারিত আলোচনা করা হলো:

১. টোকেনাইজার লোড করা

```
from transformers import AutoTokenizer
checkpoint = "distilbert-base-uncased-finetuned-sst-2-english"
tokenizer = AutoTokenizer.from_pretrained(checkpoint)
```

এখানে `AutoTokenizer` ব্যবহার করে একটি নির্দিষ্ট মডেলের (DistilBERT) জন্য আগে থেকে ট্রেইন করা টোকেনাইজারটি লোড করা হয়েছে।

২. সরাসরি টোকেনাইজেশন

আগে আমরা আলাদাভাবে টোকেন তৈরি বা আইডি-তে রূপান্তর করতাম, কিন্তু এখন সরাসরি `tokenizer(sequence)` কল করলেই এটি ইনপুট আইডি (`input_ids`) এবং অ্যাটেনশন মাস্ক (`attention_mask`) তৈরি করে দেয়। এটি একক বাক্য বা একাধিক বাক্যের লিস্ট—উভয়ই প্রসেস করতে পারে।

৩. প্যাডিং **(Padding)** এর বিভিন্ন নিয়ম

একাধিক বাক্যকে সমান দৈর্ঘ্যের করার জন্য `padding` প্যারামিটার ব্যবহার করা হয়:

- **`padding="longest"`:** ব্যাচের মধ্যে থাকা সবচেয়ে বড় বাক্যের সমান দৈর্ঘ্য করবে।
- **`padding="max_length"`:** মডেলের সর্বোচ্চ ক্ষমতা (যেমন ৫১২) অনুযায়ী প্যাডিং করবে।
- **`max_length=8`:** আপনার নির্দিষ্ট করে দেওয়া দৈর্ঘ্য পর্যন্ত প্যাডিং করবে।

৪. ট্রাঙ্কেশন **(Truncation)** বা ছোট করা

বাক্য যদি অনেক বড় হয়, তবে `truncation=True` ব্যবহার করে সেটিকে মডেলের সর্বোচ্চ সীমা বা আপনার দেওয়া `max_length` অনুযায়ী ছেঁটে ছোট করা যায়।

৫. টেনসর **(Tensor)** রিটার্ন করা

মডেল সরাসরি পাইথন লিস্ট প্রসেস করতে পারে না, তাই টেনসর প্রয়োজন হয়:

- **`return_tensors="pt"`:** এটি PyTorch টেনসর রিটার্ন করে।
- **`return_tensors="np"`:** এটি NumPy অ্যারে রিটার্ন করে।

৬. স্পেশাল টোকেন **(Special Tokens)**

টোকেনাইজার নিজে থেকেই বাক্যের শুরুতে `[CLS]` (শ্রেণীবিন্যাসের জন্য) এবং শেষে `[SEP]` (বাক্য আলাদা করার জন্য) টোকেন যোগ করে দেয়। কারণ মডেলটি যখন প্রি-ট্রেইন করা হয়েছিল, তখন এই টোকেনগুলো ব্যবহার করা হয়েছিল। তাই সঠিক ফলাফলের জন্য এগুলো প্রয়োজন।

৭. সব ধাপের সমন্বয় (Final Code)

সবশেষে, টোকেনাইজার থেকে মডেল পর্যন্ত পুরো প্রক্রিয়াটি একবারে দেখানো হয়েছে:

```
# ১. টোকেন তৈরি, প্যাডিং, ট্রাঙ্কেশন এবং PyTorch টেনসর হিসেবে রিটার্ন
tokens = tokenizer(sequences, padding=True, truncation=True, return_tensors="pt")

# ২. মডেলে টেনসরগুলো ইনপুট হিসেবে পাঠানো
output = model(**tokens)
```

এখানে `**tokens` ব্যবহার করে সব ইনপুট আইডি এবং মাস্ক সরাসরি মডেলে পাঠিয়ে আউটপুট বের করা হয়েছে।

সংক্ষেপে, এই পদ্ধতিটি ব্যবহার করলে আপনাকে আলাদাভাবে প্রতিটি ধাপ (যেমন প্যাডিং বা মাস্ক তৈরি) হাতে কলমে করতে হয় না; টোকেনাইজার নিজেই সব সামলে নেয়।