```
# %%capture
!pip install -q transformers==4.44.2 datasets==2.21.0 evaluate==0.4.2 accelerate==0.34.2 sentencepiece==0.2.0
```

```
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 43.7/43.7 kB 2.3 MB/s eta 0:00:00
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 9.5/9.5 MB 77.3 MB/s eta 0:00:00
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 527.3/527.3 kB 26.7 MB/s eta 0:00:00
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 84.1/84.1 kB 7.6 MB/s eta 0:00:00
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 324.4/324.4 kB 16.7 MB/s eta 0:00:00
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1.3/1.3 MB 62.1 MB/s eta 0:00:00
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 177.6/177.6 kB 12.1 MB/s eta 0:00:00
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 3.6/3.6 MB 76.4 MB/s eta 0:00:00
ERROR: pip's dependency resolver does not currently take into account all the packages that are installed. This behaviour is the sou
gcsfs 2025.3.0 requires fsspec==2025.3.0, but you have fsspec 2024.6.1 which is incompatible.
```

```
!pip install -U fsspec gcsfs
```

```
Requirement already satisfied: fsspec in /usr/local/lib/python3.12/dist-packages (2024.6.1)
Collecting fsspec
  Downloading fsspec-2025.9.0-py3-none-any.whl.metadata (10 kB)
Requirement already satisfied: gcsfs in /usr/local/lib/python3.12/dist-packages (2025.3.0)
Collecting gcsfs
  Downloading gcsfs-2025.9.0-py2.py3-none-any.whl.metadata (2.1 kB)
Requirement already satisfied: aiohttp!=4.0.0a0,!=4.0.0a1 in /usr/local/lib/python3.12/dist-packages (from gcsfs) (3.13.1)
Requirement already satisfied: decorator>4.1.2 in /usr/local/lib/python3.12/dist-packages (from gcsfs) (4.4.2)
Requirement already satisfied: google-auth>=1.2 in /usr/local/lib/python3.12/dist-packages (from gcsfs) (2.38.0)
Requirement already satisfied: google-auth-oauthlib in /usr/local/lib/python3.12/dist-packages (from gcsfs) (1.2.2)
Requirement already satisfied: google-cloud-storage in /usr/local/lib/python3.12/dist-packages (from gcsfs) (2.19.0)
Requirement already satisfied: requests in /usr/local/lib/python3.12/dist-packages (from gcsfs) (2.32.4)
Requirement already satisfied: aiohappyeyeballs>=2.5.0 in /usr/local/lib/python3.12/dist-packages (from aiohttp!=4.0.0a0,!=4.0.0a1->g
Requirement already satisfied: aiosignal>=1.4.0 in /usr/local/lib/python3.12/dist-packages (from aiohttp!=4.0.0a0,!=4.0.0a1->gcsfs)
Requirement already satisfied: attrs>=17.3.0 in /usr/local/lib/python3.12/dist-packages (from aiohttp!=4.0.0a0,!=4.0.0a1->gcsfs) (25
Requirement already satisfied: frozenlist>=1.1.1 in /usr/local/lib/python3.12/dist-packages (from aiohttp!=4.0.0a0,!=4.0.0a1->gcsfs)
Requirement already satisfied: multidict<7.0,>=4.5 in /usr/local/lib/python3.12/dist-packages (from aiohttp!=4.0.0a0,!=4.0.0a1->gcsf
Requirement already satisfied: propcache>=0.2.0 in /usr/local/lib/python3.12/dist-packages (from aiohttp!=4.0.0a0,!=4.0.0a1->gcsfs)
Requirement already satisfied: yarl<2.0,>=1.17.0 in /usr/local/lib/python3.12/dist-packages (from aiohttp!=4.0.0a0,!=4.0.0a1->gcsfs)
Requirement already satisfied: cachetools<6.0,>=2.0.0 in /usr/local/lib/python3.12/dist-packages (from google-auth>=1.2->gcsfs) (5.5
Requirement already satisfied: pyasn1-modules>=0.2.1 in /usr/local/lib/python3.12/dist-packages (from google-auth>=1.2->gcsfs) (0.4.
Requirement already satisfied: rsa<5,>=3.1.4 in /usr/local/lib/python3.12/dist-packages (from google-auth>=1.2->gcsfs) (4.9.1)
Requirement already satisfied: requests-oauthlib>=0.7.0 in /usr/local/lib/python3.12/dist-packages (from google-auth-oauthlib->gcsfs
Requirement already satisfied: google-api-core<3.0.0dev,>=2.15.0 in /usr/local/lib/python3.12/dist-packages (from google-cloud-storag
Requirement already satisfied: google-cloud-core<3.0dev,>=2.3.0 in /usr/local/lib/python3.12/dist-packages (from google-cloud-storage
Requirement already satisfied: google-resumable-media>=2.7.2 in /usr/local/lib/python3.12/dist-packages (from google-cloud-storage->g
Requirement already satisfied: google-crc32c<2.0dev,>=1.0 in /usr/local/lib/python3.12/dist-packages (from google-cloud-storage->gcs
Requirement already satisfied: charset_normalizer<4,>=2 in /usr/local/lib/python3.12/dist-packages (from requests->gcsfs) (3.4.4)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.12/dist-packages (from requests->gcsfs) (3.11)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.12/dist-packages (from requests->gcsfs) (2.5.0)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.12/dist-packages (from requests->gcsfs) (2025.10.5)
Requirement already satisfied: typing-extensions>=4.2 in /usr/local/lib/python3.12/dist-packages (from aiosignal>=1.4.0->aiohttp!=4.0
Requirement already satisfied: googleapis-common-protos<2.0.0,>=1.56.2 in /usr/local/lib/python3.12/dist-packages (from google-api-co
Requirement already satisfied: protobuf!=3.20.0,!=3.20.1,!=4.21.0,!=4.21.1,!=4.21.2,!=4.21.3,!=4.21.4,!=4.21.5,<7.0.0,>=3.19.5 in /u
Requirement already satisfied: proto-plus<2.0.0,>=1.22.3 in /usr/local/lib/python3.12/dist-packages (from google-api-core<3.0.0dev,>=
Requirement already satisfied: pyasn1<0.7.0,>=0.6.1 in /usr/local/lib/python3.12/dist-packages (from pyasn1-modules>=0.2.1->google-a
Requirement already satisfied: oauthlib>=3.0.0 in /usr/local/lib/python3.12/dist-packages (from requests-oauthlib>=0.7.0->google-aut
Downloading fsspec-2025.9.0-py3-none-any.whl (199 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 199.3/199.3 kB 4.4 MB/s eta 0:00:00
Downloading gcsfs-2025.9.0-py2.py3-none-any.whl (36 kB)
Installing collected packages: fsspec, gcsfs
  Attempting uninstall: fsspec
    Found existing installation: fsspec 2024.6.1
    Uninstalling fsspec-2024.6.1:
      Successfully uninstalled fsspec-2024.6.1
  Attempting uninstall: gcsfs
    Found existing installation: gcsfs 2025.3.0
    Uninstalling gcsfs-2025.3.0:
      Successfully uninstalled gcsfs-2025.3.0
ERROR: pip's dependency resolver does not currently take into account all the packages that are installed. This behaviour is the sou
datasets 2.21.0 requires fsspec[http]<=2024.6.1,>=2023.1.0, but you have fsspec 2025.9.0 which is incompatible.
Successfully installed fsspec-2025.9.0 gcsfs-2025.9.0
```

```
import numpy as np
from typing import Dict, List, Optional

import torch
from datasets import load_dataset, DatasetDict, ClassLabel
from transformers import (
```

```
    AutoTokenizer,
    AutoConfig,
    AutoModelForSequenceClassification,
    DataCollatorWithPadding,
    Trainer,
    TrainingArguments,
    set_seed,
)
import evaluate

set_seed(42)
print("Torch:", torch.__version__, "| CUDA:", torch.cuda.is_available())
```

```
Torch: 2.8.0+cu126 | CUDA: True
```

```
BASE_MODEL = "ai4bharat/IndicBERTv2-MLM-only"  # IndicBERT v2 encoder (hi + bn supported)
MAX_LEN = 256
NUM_LABELS = 3   # entailment, neutral, contradiction
BATCH_SIZE = 8   # per-device
GRAD_ACCUM = 2   # effective batch 16
LR = 2e-5
EPOCHS = 3
FP16 = True
```

```
# Hindi from IndicXNLI (must specify the 'hi' config)
indicxnli_hi = load_dataset("Divyanshu/indicxnli", "hi")

# Build Hindi train/dev
# Most mirrors expose 'train' and 'validation' splits. If not, we create a small dev from train.
if "train" in indicxnli_hi:
    train_hi = indicxnli_hi["train"]
else:
    # Extremely rare fallback: some mirrors only have a single split
    tmp = indicxnli_hi["validation"]
    train_hi = tmp.train_test_split(test_size=0.2, seed=42)["train"]

if "validation" in indicxnli_hi:
    dev_hi = indicxnli_hi["validation"]
else:
    dev_hi = train_hi.train_test_split(test_size=0.05, seed=42)["test"]

# Bangla test from xnli_bn (rename to match field names used for Hindi)
xnli_bn = load_dataset("csebuetnlp/xnli_bn")
test_bn = xnli_bn["test"].rename_columns({"sentence1": "premise", "sentence2": "hypothesis"})

# Quick sanity checks
print("HI splits:", list(indicxnli_hi.keys()))
print("HI example:", {k: train_hi[0][k] for k in ["premise","hypothesis","label"]})
print("BN example:", {k: test_bn[0][k] for k in ["premise","hypothesis","label"]})
```

```
/usr/local/lib/python3.12/dist-packages/huggingface_hub/utils/_auth.py:94: UserWarning:
The secret `HF_TOKEN` does not exist in your Colab secrets.
To authenticate with the Hugging Face Hub, create a token in your settings tab (https://huggingface.co/settings/tokens), set it as s
You will be able to reuse this secret in all of your notebooks.
Please note that authentication is recommended but still optional to access public models or datasets.
  warnings.warn(
Downloading data: 100%                                  74.6M/74.6M [00:00<00:00, 148MB/s]
Downloading data: 100%                                  477k/477k [00:00<00:00, 3.49MB/s]
Downloading data: 100%                                  239k/239k [00:00<00:00, 1.86MB/s]
Generating train split: 100%                            392702/392702 [00:01<00:00, 160824.65 examples/s]
Generating test split: 100%                             5010/5010 [00:00<00:00, 91148.88 examples/s]
Generating validation split: 100%                       2490/2490 [00:00<00:00, 51525.81 examples/s]
Downloading data: 100%                                  74.6M/74.6M [00:00<00:00, 211MB/s]
Downloading data: 100%                                  480k/480k [00:00<00:00, 1.84MB/s]
Downloading data: 100%                                  242k/242k [00:00<00:00, 1.01MB/s]
Generating train split: 100%                            381449/381449 [00:02<00:00, 154455.03 examples/s]
Generating test split: 100%                             4895/4895 [00:00<00:00, 89070.94 examples/s]
Generating validation split: 100%                       2419/2419 [00:00<00:00, 51586.44 examples/s]
HI splits: ['train', 'test', 'validation']
HI example: {'premise': 'अवधारणात्मक रूप से क्रीम स्किमिंग के दो बुनियादी आयाम हैं-उत्पाद और भूगोल।', 'hypothesis': 'उत्पाद और भूगोल क्रीम स्किमिंग क
BN example: {'premise': 'আসলে, আমি এমনকি এই বিষয়ে চিন্তাও করিনি, কিন্তু আমি এত হতাশ হয়ে পড়েছিলাম যে, শেষ পর্যন্ত আমি আবার তার সঙ্গে ব
```

```python
label_names = ["entailment", "neutral", "contradiction"]
label_to_id = {n: i for i, n in enumerate(label_names)}

def normalize_label_column(ds):
    # If labels are ClassLabel, map indices via names; else map strings directly.
    if isinstance(ds.features["label"], ClassLabel):
        names = ds.features["label"].names
        remap = {names.index(n): label_to_id[n] for n in names if n in label_to_id}
        def _map(ex):
            ex["label"] = remap.get(ex["label"], ex["label"])
            return ex
        return ds.map(_map)
    else:
        def _map(ex):
            if isinstance(ex["label"], str):
                ex["label"] = label_to_id[ex["label"]]
            return ex
        return ds.map(_map)

train_hi = normalize_label_column(train_hi)
dev_hi   = normalize_label_column(dev_hi)

# xnli_bn labels are ["contradiction","entailment","neutral"]; remap by name
def remap_bn(ex):
    names = test_bn.features["label"].names if isinstance(test_bn.features["label"], ClassLabel) else None
    if names is not None:
        ex["label"] = label_to_id[names[ex["label"]]]
    return ex

test_bn = test_bn.map(remap_bn)

raw = DatasetDict(train=train_hi, validation=dev_hi, test=test_bn)
raw
```

```
Map: 100%                                                    392702/392702 [00:31<00:00, 23108.29 examples/s]
Map: 100%                                                    2490/2490 [00:00<00:00, 20713.50 examples/s]
Map: 100%                                                    4895/4895 [00:00<00:00, 7346.81 examples/s]
DatasetDict({
    train: Dataset({
        features: ['premise', 'hypothesis', 'label'],
        num_rows: 392702
    })
    validation: Dataset({
        features: ['premise', 'hypothesis', 'label'],
        num_rows: 2490
    })
    test: Dataset({
        features: ['premise', 'hypothesis', 'label'],
        num_rows: 4895
    })
})
```

```
from transformers import AutoTokenizer, AutoConfig, AutoModelForSequenceClassification

tok = AutoTokenizer.from_pretrained(BASE_MODEL, use_fast=True)

cfg = AutoConfig.from_pretrained(
    BASE_MODEL,
    num_labels=NUM_LABELS,
    problem_type="single_label_classification",
)

model = AutoModelForSequenceClassification.from_pretrained(BASE_MODEL, config=cfg)
```

```
tokenizer_config.json: 100%                                  51.0/51.0 [00:00<00:00, 5.94kB/s]
tokenizer.json: 100%                                         7.75M/7.75M [00:00<00:00, 10.7MB/s]
special_tokens_map.json: 100%                                125/125 [00:00<00:00, 15.6kB/s]
/usr/local/lib/python3.12/dist-packages/transformers/tokenization_utils_base.py:1601: FutureWarning: `clean_up_tokenization_spaces` u
  warnings.warn(
config.json: 100%                                            639/639 [00:00<00:00, 65.1kB/s]
pytorch_model.bin: 100%                                      1.12G/1.12G [00:22<00:00, 75.7MB/s]
Some weights of BertForSequenceClassification were not initialized from the model checkpoint at ai4bharat/IndicBERTv2-MLM-only and a
You should probably TRAIN this model on a down-stream task to be able to use it for predictions and inference.
```

```
def preprocess(batch):
    return tok(
        batch["premise"],
        batch["hypothesis"],
        truncation=True,
        max_length=MAX_LEN,
    )

cols_to_keep = {"premise", "hypothesis", "label"}
cols_to_remove = [c for c in raw["train"].column_names if c not in cols_to_keep]

tokenized = raw.map(preprocess, batched=True, remove_columns=cols_to_remove)
data_collator = DataCollatorWithPadding(tokenizer=tok)

for split in ["train", "validation", "test"]:
    print(split, tokenized[split][0].keys())
```

```
Map: 100%                                                    392702/392702 [00:57<00:00, 3026.16 examples/s]
Map: 100%                                                    2490/2490 [00:00<00:00, 3654.61 examples/s]
Map: 100%                                                    4895/4895 [00:01<00:00, 4640.04 examples/s]
train dict_keys(['premise', 'hypothesis', 'label', 'input_ids', 'token_type_ids', 'attention_mask'])
validation dict_keys(['premise', 'hypothesis', 'label', 'input_ids', 'token_type_ids', 'attention_mask'])
test dict_keys(['premise', 'hypothesis', 'label', 'input_ids', 'token_type_ids', 'attention_mask'])
```

```
!pip install -q scikit-learn
```

```
from sklearn.metrics import accuracy_score, f1_score
import numpy as np

def compute_metrics(eval_pred):
    logits, labels = eval_pred
    preds = np.argmax(logits, axis=-1)
    return {
        "accuracy": float(accuracy_score(labels, preds)),
        "macro_f1": float(f1_score(labels, preds, average="macro")),
    }
```

```
from transformers import TrainingArguments, Trainer
import torch

out_dir = "indicbertv2_hi_to_bn_nli"

args = TrainingArguments(
    output_dir=out_dir,
    evaluation_strategy="epoch",
    save_strategy="epoch",
    save_total_limit=1,              # keep disk use low
    learning_rate=LR,
    per_device_train_batch_size=BATCH_SIZE,
    per_device_eval_batch_size=BATCH_SIZE,
    gradient_accumulation_steps=GRAD_ACCUM,
    num_train_epochs=EPOCHS,
    logging_steps=50,
    load_best_model_at_end=True,
    metric_for_best_model="macro_f1",
    greater_is_better=True,
    fp16=FP16,
    report_to="none",
    dataloader_pin_memory=torch.cuda.is_available(),  # avoids warning on CPU
)

trainer = Trainer(
    model=model,
    args=args,
    train_dataset=tokenized["train"],
    eval_dataset=tokenized["validation"],
    tokenizer=tok,
    data_collator=data_collator,
    compute_metrics=compute_metrics,
)

# ---------------------------------------------------------------------
# Fix for ValueError: "non-contiguous tensor" when saving
# ---------------------------------------------------------------------
def make_model_contiguous(model):
    """Ensures all model weights are contiguous in memory."""
    with torch.no_grad():
        for name, param in model.named_parameters():
            if not param.is_contiguous():
                param.data = param.contiguous()

make_model_contiguous(trainer.model)

# ---------------------------------------------------------------------
# Train model
# ---------------------------------------------------------------------
train_result = trainer.train()

# ---------------------------------------------------------------------
# Optionally save model manually to ensure success
# ---------------------------------------------------------------------
trainer.model.save_pretrained("indicbertv2_hi_to_bn_final")
tok.save_pretrained("indicbertv2_hi_to_bn_final")
```

```
/usr/local/lib/python3.12/dist-packages/transformers/training_args.py:1525: FutureWarning: `evaluation_strategy` is deprecated and w
  warnings.warn(
/usr/local/lib/python3.12/dist-packages/accelerate/accelerator.py:494: FutureWarning: `torch.cuda.amp.GradScaler(args...)` is deprec
  self.scaler = torch.cuda.amp.GradScaler(**kwargs)
```

[69981/73632 4:03:24 < 12:41, 4.79 it/s, Epoch 2.85/3]

Epoch	Training Loss	Validation Loss	Accuracy	Macro F1
1	0.570800	0.576954	0.773494	0.773098