# Steps to run the code

(Run in google colab)

1. !python tokenizer\_utils.py train-byte-level iwslt14 10000 1
2. !mkdir ckpts
3. !pip install blobfile
4. !pip install mpi4py
5. !pip install bert-score
6. Place the files train\_selector8\_embeddings.pkl,valid\_selector8\_embeddings.pkl,test\_selctor8\_embeddings.pkl in the folder /data/iwslt14/

(these files are present in folder /RIGHT/ )

(\* do this only after step 1 is done)

1. Run the following command

import os

import subprocess

# Set parameters

GPU = "0" # Change as needed

NUM\_GPUS = 1

LOSS\_FUNC = "uniform"

SRC = "de" # Change as needed

TGT = "en" # Change as needed

LR = 0.0001

SEQ\_LEN = 8

WARMUP = 1500

SCHEDULE\_UPDATE\_STRIDE = 200

DSET = "iwslt\_ckpts"

UPDATE\_GRANU = 20

INIT\_PRETRAINED\_MODEL = "False"

USE\_PRETRAINED\_EMBEDDINGS = "False"

FREEZE\_EMBEDDINGS = "False"

LR\_ANNEAL\_STEPS = 3600

DIFFUSION\_STEPS = 200

NOISE\_SCHEDULE = "cosine"

BATCH\_SIZE = 128

CHECKPOINT\_PATH = f"ckpts/{DSET}/iwslt14\_{SEQ\_LEN}\_{LR}\_{DIFFUSION\_STEPS}\_{LR\_ANNEAL\_STEPS}\_{WARMUP}\_schegran{SCHEDULE\_UPDATE\_STRIDE}\_src{SRC}\_tgt{TGT}"

TRAIN\_TXT\_PATH = "./data/iwslt14/train"

VAL\_TXT\_PATH = "./data/iwslt14/valid"

IN\_CHANNELS = 512

WEIGHT\_DECAY = 0.0

SEED = 10708

DROPOUT = 0.3

NUM\_HEADS = 8

CONFIG\_NAME = "facebook/bart-base"

NOTES = "iwslt14 training with noise schedule and self condition"

# Create directories

os.makedirs(CHECKPOINT\_PATH, exist\_ok=True)

os.makedirs(os.path.join(CHECKPOINT\_PATH, "log"), exist\_ok=True)

# Prepare arguments

ARGS = [

"--checkpoint\_path", CHECKPOINT\_PATH,

"--save\_interval", str(WARMUP),

"--lr", str(LR),

"--batch\_size", str(BATCH\_SIZE),

"--src", SRC,

"--tgt", TGT,

"--diffusion\_steps", str(DIFFUSION\_STEPS),

"--noise\_schedule", NOISE\_SCHEDULE,

"--sequence\_len", str(SEQ\_LEN),

"--seed", str(SEED),

"--weight\_decay", str(WEIGHT\_DECAY),

"--predict\_xstart", "True",

"--train\_txt\_path", TRAIN\_TXT\_PATH,

"--dataset", "iwslt14",

"--val\_txt\_path", VAL\_TXT\_PATH,

"--config\_name", CONFIG\_NAME,

"--init\_pretrained", INIT\_PRETRAINED\_MODEL,

"--freeze\_embeddings", FREEZE\_EMBEDDINGS,

"--use\_pretrained\_embeddings", USE\_PRETRAINED\_EMBEDDINGS,

"--notes", NOTES

]

if LR\_ANNEAL\_STEPS == 0:

LR\_ANNEAL\_STEPS = 100

DEBUG = True

else:

DEBUG = False

ARGS += ["--lr\_anneal\_steps", str(LR\_ANNEAL\_STEPS)]

if DEBUG:

ARGS.append("--debug")

ARGS += [

"--encoder\_layers", "6",

"--decoder\_layers", "6",

"--num\_heads", "8",

"--in\_channel", "512",

"--out\_channel", "512",

"--num\_channels", "2048",

"--sequence\_len\_src", "128",

"--warmup", str(WARMUP),

"--schedule\_sampler", LOSS\_FUNC,

"--loss\_update\_granu", str(UPDATE\_GRANU),

"--schedule\_update\_stride", str(SCHEDULE\_UPDATE\_STRIDE)

]

# Set CUDA device1

os.environ["CUDA\_VISIBLE\_DEVICES"] = GPU

# Execute the command

subprocess.run(["mpiexec", "--allow-run-as-root", "-n", str(NUM\_GPUS), "python", "-u", "main.py"] + ARGS, capture\_output=True, text=True)

1. Run the following command

# Define parameters

MODEL\_NAME = "/content/ckpts/iwslt\_ckpts/iwslt14\_8\_0.0001\_200\_3000\_1500\_schegran200\_srcde\_tgten/ema\_0.9999\_003600.pt" # Replace with your actual model name

OUT\_DIR = "your\_output\_directory" # Replace with your desired output directory

SCHEDULE\_PATH = "/content/ckpts/iwslt\_ckpts/iwslt14\_8\_0.0001\_200\_3600\_1500\_schegran200\_srcde\_tgten/alpha\_cumprod\_step\_800.npy" # Replace with your actual schedule path

SEED = 10708 # or any seed you prefer

GEN\_BY\_Q = "False"

GEN\_BY\_MIX = "False"

MIX\_PROB = 0

MIX\_PART = 1

TOP\_P = -1

CLAMP = "no\_clamp"

BATCH\_SIZE = 100

DIFFUSION\_STEPS = 200

NUM\_SAMPLES = -1

SEQ\_LEN = 8

VAL\_TXT = "./data/iwslt14/test" # Ensure this path exists in your environment

# Execute the command

!python -u inference\_main.py \

--model\_name\_or\_path {MODEL\_NAME} \

--sequence\_len\_src 128 \

--batch\_size {BATCH\_SIZE} \

--num\_samples {NUM\_SAMPLES} \

--top\_p {TOP\_P} \

--time\_schedule\_path {SCHEDULE\_PATH} \

--seed {SEED} \

--val\_txt\_path {VAL\_TXT} \

--generate\_by\_q {GEN\_BY\_Q} \

--generate\_by\_mix {GEN\_BY\_MIX} \

--out\_dir {OUT\_DIR} \

--diffusion\_steps {DIFFUSION\_STEPS} \

--clamp {CLAMP} \

--sequence\_len {SEQ\_LEN} \

--generate\_by\_mix\_prob {MIX\_PROB} \

--generate\_by\_mix\_part {MIX\_PART}

1. Two files will be generated
   1. Contains decoded outputs
   2. Contains undecoded outputs
2. Place this path(decoded) in bert\_score.py file and run bert\_score.py file