ECE4730J Advanced Embedded System Mid-Check

ELMA: Encrypted Offloading for Embedded NLP Applications

Shuocheng Chen, Yihua Liu, Yiming Ju

UM-SJTU Joint Institute

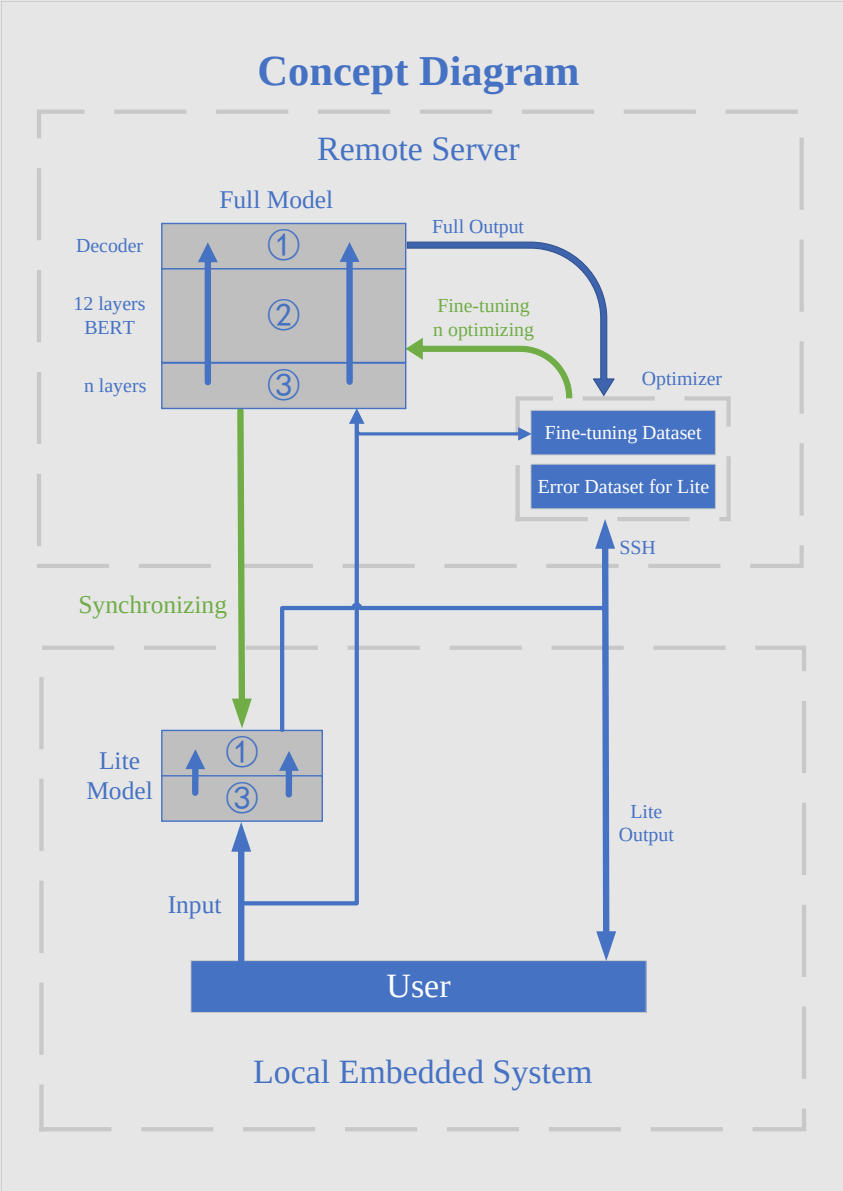
November 8, 2021

Overview

Tasks:

- Run `ALBERT` on remote servers.
- Fine-tuning `ALBERT` on remote servers.
- Run the lite model edgeBERT
- Apply `edgeBERT` on Jetson TX2.

Methodology



ALBERT on Jetson TX2

Try to run ALBERT with tensorflow==1.15.2 on Jetson TX2.

Difficulties

AArch64 (ARMv8) platform is extremely poorly supported - no APT, no pip, even no wheels. Even Raspberry Pi (ARMv7) has better support. Most of the packages have to be compiled and built manually. Network was also a head-scratching problem.

- Successfully installed JetPack 4.6 environment, which includes CUDA==10.2.
- Successfully built bazel==0.26.1 with CUDA support.
- Successfully set up SSH/SFTP/SCP connections and proxies for AArch64 (ARMv8) platform.
- Failed to build tensorflow==1.15.2 with cuDNN==8.2.1 and TensorRT==8.0.1-1+cuda10.2 because some incompatibilities are unresolvable.

ALBERT on Jetson TX2

Difficulties

Building `tensorflow` using `bazel` costs extremely large disk space. It costs the remaining 31% of the embedded disk `/dev/mmcblk0p1` or 2% of the external disk (8.5 GiB), while `/dev/mmcblk0p1` only has 28768292 1K-blocks (28,094 MiB or 27.4 GiB). Beyond doubt it will cost disk space much more than this value.

When building caches occupy the whole disk space, Jetson TX2 system corrupted, which is dangerous. Referring to https://blog.csdn.net/weixin_48695448/article/details/117337766, it costs a lot of time to repair.

Solution

Use an external disk. Mount the disk, and specify the output path of `bazelbuild`.

ALBERT on Jetson TX2

All the experiences are summarized in https://blog.csdn.net/yihuajack/article/details/121045347. It might be easier to build a CPU-only version, but it is meaningless.

Plan

We have to use TensorFlow 2 because migrating ALBERT from tf1 to tf2 is much more easier than migrating tensorflow==1.15.2 from cuDNN==7 to cuDNN==8 and from TensorRT==7 to TensorRT==8.

Plan

Accelerate the progress of building the application. The application takes speech audio as input, does speech recognition, converts speech to text, does NLP question-answering and computation offloading, and takes answers as output.

ALBERT for TensorFlow 2

Current solutions:

- bert-for-tf2:
- ALBERT-TF2.0: Lack of maintenance, last commit is on Dec 9, 2019.
- Issues of ALBERT.

Development approaches:

- Start from ALBERT and put compability patches on it.
- Start from bert-to-tf2 and find possible ways to customize layers.

Albert

Preparation

1. Configure the environment on the server
2. Choose the pretrained model from https://tfhub.dev/google/albert_large/3
3. Download the data set SQuAD from https://rajpurkar.github.io/SQuAD-explorer/

Albert

Result

```
I0922 11:46:38.663871 140308634334976 run_squad_v2.py:505]
***** Final Eval results *****
INFO:tensorflow: exact = 50.09685841825992
I0922 11:46:38.663987 140308634334976 run_squad_v2.py:507] exact = 50.09685841825992
INFO:tensorflow: f1 = 50.11359538016659
I0922 11:46:38.664040 140308634334976 run_squad_v2.py:507] f1 = 50.11359538016659
INFO:tensorflow: null_score_diff_threshold = -1.230899453163147
I0922 11:46:38.664077 140308634334976 run_squad_v2.py:507]
null_score_diff_threshold = -1.230899453163147
INFO:tensorflow: total = 11873
I0922 11:46:38.664113 140308634334976 run_squad_v2.py:507] total = 11873
```

The result is too low compared to the F1 score in the paper about 85.

Albert

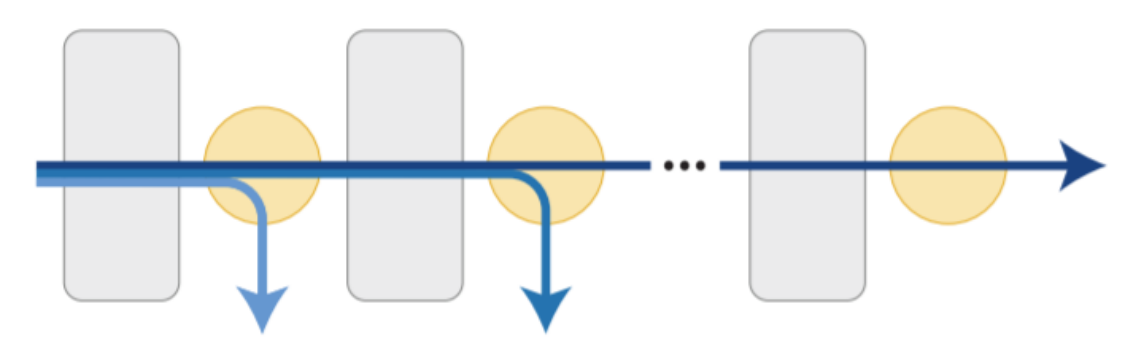
Problems

1. F1 score is too low compared to the F1 score in the paper
2. The version of tensorflow is not consistent with the version in the embedded device

Plans

1. Solve the problems above
2. Realize the function of updating the data set with the unsolvable questions from the lite model

Early Exit



- Grey blocks are transformer layers
- Orange circles are classification layers
- Blue arrows are possible exits

Early Exit

Advantage

- Prevent overfitting
- Reduce computation

Characteristic

- Match linguistically complex sentences with larger models and simple sentences with smaller models
- A lightweight classifier at the output of the transformer layer
- Threshold E_T
- Entropy

$$H(x) = -\sum p(x) log p(x) = ln(\sum_{k=1}^{n} e^{x_k}) - \frac{\sum_{k=1}^{n} x_k e^{x_k}}{\sum_{k=1}^{n} x_k}$$

Early Exit

Algorithm

for layer i from 1 to n do
    if $H(z_i) < E_T$ then
        return z_i
    end if
end for
if the question is solved
    return z_n
else
    return unsolvable question to the server

*z_i is the output of each layer

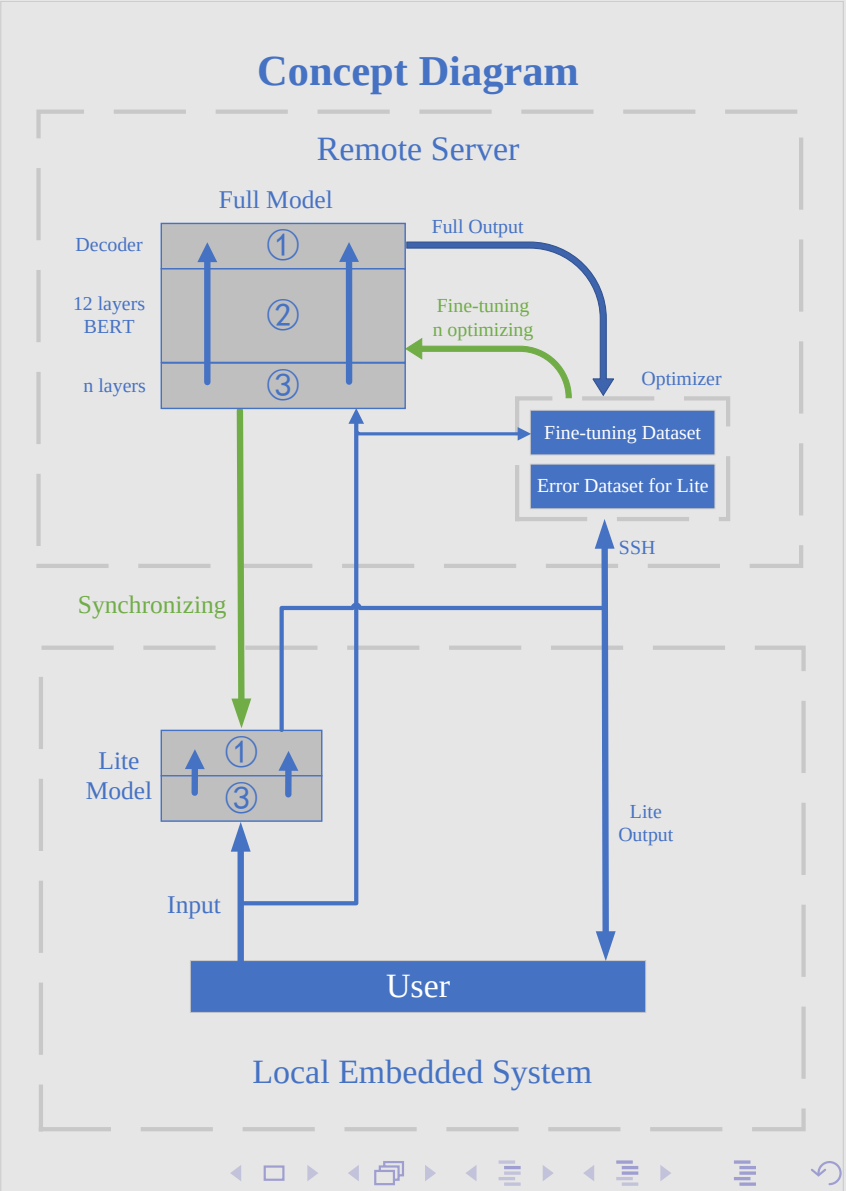
Early Exit

Plan

- Realize the algorithm
- Apply the model to the data set SQuAD
- Test the different number of layers

Future plan

- Synchronize the lite model with the full model
- Automatically update the data set on the server with the unsolvable questions on lite model
- Migrate the model to the embedded device



Thanks!