@ mktshhr updated at 2018-10-21

Run Tensorflow with THETA V

Java Android

4

In the beginning

Tensorflow has a sample that runs on Android, and is also featured on the following sites:
https://qiita.com/icchi_h/items/a1df9f27569714edfc5e
Here, Tensorflow is operated with the omnidirectional camera THETA V that is equipped with Android.
(2018/8/18: Merged and added articles that were divided into the front and back part)

Development environment

- RICOH THETA V, firmware ver.2.40.2 (Android 7.1.1)
- Moto G5 Plus, Android 7.0
- Android Studio 3.1.4
- Build #AI-173.4907809, built on July 24, 2018
- JRE: 1.8.0_152-release-1024-b01 x86_64
- JVM: OpenJDK 64-Bit Server VM by JetBrains s.r.o
- Mac OS X 10.13.6

procedure

Source code acquisition (Tensorflow, PluginSDK)

Get tensorflow from github and prepare the source code. Here, the fork of https://github.com/tensorflow/tensorflow (commit: 26353f9b51091312e7097143aee9c2d05e2011fd) is placed at https://github.com/mktshhr/tensorflow-theta and updated for THETA V. The tag "Qiita 20180818" is the source code corresponding to this article. Among these, https://github.com/mktshhr/tensorflow-theta/tree/Qiita20180818/tensorflow/examples/android is the scope of change.
Once you have the source code, you can try opening / tensorflow-theta / examples / android in Android Studio. The changes described below have already been applied.

```
git clone https://github.com/mktshhr/tensorflow-theta.git
cd ./tensorflow-theta/examples/android
```

Open in Android Studio

Open the android folder in Android Studio.
Follow the screen by clicking "Add Google Maven repository and sync project".
Rewrite the 45th line of build.gradle from "bazel" to "none".

```
// set to 'bazel', 'cmake', 'makefile', 'none'
def nativeBuildSystem = 'none'
```

Similarly, line 194 of build.gradle is rewritten from "compile" to "implementation".

```
dependencies {
    if (nativeBuildSystem == 'cmake' || nativeBuildSystem == 'none') {
        implementation 'org.tensorflow:tensorflow-android:+'
    }
}
```

Update "Sync Now".

Move on Android smartphone

It works on Android smartphones (Moto G5 Plus, Android 7.0) with the changes so far. You can select "Run"-"Debug 'Tensorflow Android'", build it, and run it on your Android s martphone. (In the case of THETA V, the camera shuts down as it is pipipipipi.)

TensorFlow Demo consists of four activities (apps): TF Classify, TF Detect, TF Stylize, and TF Speech.

- TF Classify: Classification into 1000 categories.
- TF Detect: Object recognition using YOLO.
- TF Stylize: style conversion such as painting style.
- TF Speech: Speech recognition.

Changes for THETA V

Modification of Camera common part (TF Classify, TF Detec t, TF Stylize)

- Fix CameraActivity.java

Changed to broadcast "com.theta360.plugin.ACTION_MAIN_CAMERA_CLOSE" in onCre ate of the CameraActivity class. Release THETA's camera resources and make them avail able to the Tensorflow app.

```
@Override
protected void onCreate(final Bundle savedInstanceState) {
  LOGGER.d("onCreate " + this);
  sendBroadcast(new Intent("com.theta360.plugin.ACTION_MAIN_CAMERA_CLOSE"));
  super.onCreate(null);
```

To build, add the following to the top of the CameraActivity.java file:

```
import android.content.Intent;
```

Fixed the onPreviewSizeChosen argument near line 124. I did not rotate.

```
onPreviewSizeChosen(new Size(previewSize.width, previewSize.height), 0);
```

- Fix LegacyCameraConnectionFragment.java

Change around the 99th line as follows. Here, RicMoviePreview1024 is used, but settings such as RicMoviePreview3840 are also possible.

```
            //camera.setDisplayOrientation(90);
            parameters.set("RIC_SHOOTING_MODE", "RicMoviePreview1024");
```

Changed the 109th line as follows. Swap the width and height.

```
        camera.addCallbackBuffer(new byte[ImageUtils.getYUVByteSize(s.width, s.height)])
        textureView.setAspectRatio(s.width, s.height);
```

With the above changes, the operation of TF Classify and TF Detect has been confirmed on Vysor.
The display has been corrected because the image has become vertically long, but the new Camera2 API can be used with the Android smartphone (Moto G5 Plus), but the behavior differs between the smartphone and THETA because the THETA uses the legacy Camera API. It may be.

TF Detect uses TF_OD_API (Tensorflow Object Detection API) by default for object detection.

TF Stylize fix

Once the image is a 1: 1 square image and Style applied, it is corrected back to a 2: 1 eq uilateral cylinder image.

- Make a 2: 1-> 1: 1 image for style conversion

Set to not save the aspect ratio when creating a frameToCropTransform transformation m atrix with processImage () near line 495 of StylizeActivity.java. This will allow 2: 1-> 1: 1 im ages for style conversion.

```
@Override
protected void processImage() {
  if (desiredSize != initializedSize) {
    LOGGER.i(
        "Initializing at size preview size %dx%d, stylize size %d",
        previewWidth, previewHeight, desiredSize);

    rgbFrameBitmap = Bitmap.createBitmap(previewWidth, previewHeight, Config.ARGB_8888);
    croppedBitmap = Bitmap.createBitmap(desiredSize, desiredSize, Config.ARGB_8888);
    frameToCropTransform = ImageUtils.getTransformationMatrix(
        previewWidth, previewHeight,
        desiredSize, desiredSize,
        sensorOrientation, false);
```

- Restore style converted image to 1: 1-> 2: 1

StylizeActivity.javaの520行目付近のRunnable()内で、Canvasを使ってリサイズし、stylizeI mage()でスタイル変更したcroppedBitmap画像を1:1から2:1にする。スタイル変更した2: 1のtextureCopyBitmap画像ができる。

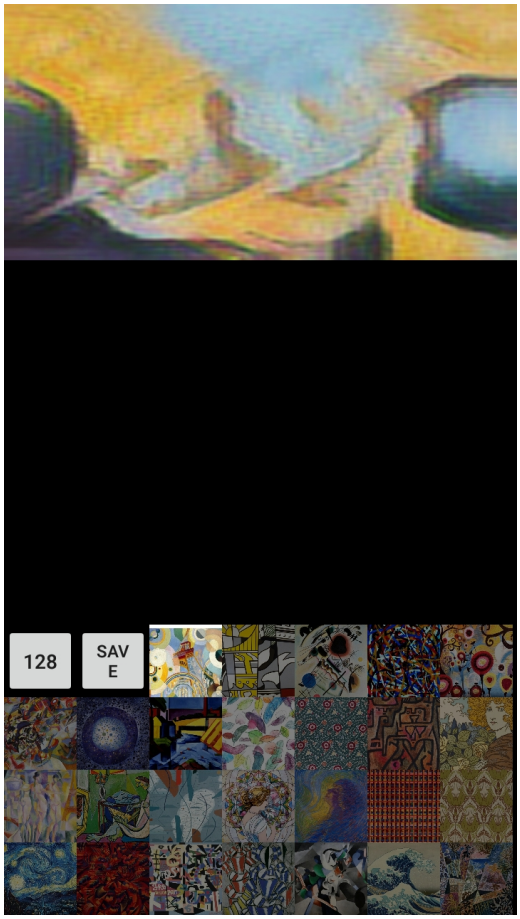
```
runInBackground(
    new Runnable() {
      @Override
      public void run() {
        cropCopyBitmap = Bitmap.createBitmap(croppedBitmap);
        final long startTime = SystemClock.uptimeMillis();
        stylizeImage(croppedBitmap);
        lastProcessingTimeMs = SystemClock.uptimeMillis() - startTime;
        textureCopyBitmap = Bitmap.createBitmap(previewWidth, previewHeight, Config.AR

        final Paint paint = new Paint();
```

```
        paint.setFilterBitmap(true);
        final Canvas canvas = new Canvas(textureCopyBitmap);
        canvas.drawBitmap(croppedBitmap, cropToFrameTransform, paint);

        if (SAVE_PREVIEW_BITMAP) {
          ImageUtils.saveBitmap(textureCopyBitmap, "stylizeImage.png");
        }
        requestRender();
        readyForNextImage();
      }
    });
```

ここまででTF Stylizeが動作するようになる。



TF Speechの修正

- SpeechActivity.java に AudioManagerの設定を追加。

```
@Override
protected void onCreate(Bundle savedInstanceState) {
  // Set up the UI.
  super.onCreate(savedInstanceState);

  AudioManager am = (AudioManager) getSystemService(Context.AUDIO_SERVICE); // for THETA
  am.setParameters("RicUseBFormat=false"); // for THETA

  setContentView(R.layout.activity_speech);
```

- MODIFY_AUDIO_SETTINGS権限を追加。

AndroidManifest.xmlで、android.permission.MODIFY_AUDIO_SETTINGS権限を追加（以下は26行目付近）。TF Speechのサンプリングレートのデフォルトは16kHzであり、THETAのデフォルトは44.1kHzであった。44.1kHzではTF Speechのフィルタがうまく動作しないようなので16kHzで動作させたが、この設定で動作させるために権限追加が必要だったと思われる。

```
<uses-permission android:name="android.permission.MODIFY_AUDIO_SETTINGS" />
```

以上の変更で、スマホ程度に音声認識できるようになった。

まとめ

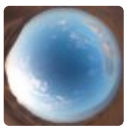
TensorflowのAndroidサンプルをTHETA上で動作させて、画像認識と音声認識を行った。TF Classifyは物体の識別器だが、全天球画像では超広角なため物体を誤判定しやすい。TF Detectは画像内の物体を探す物体検出器である。TF Stylizeはうまく2:1画像に対してスタイル適用できなかったため一旦1:1画像でスタイル適用することで回避した。TF SpeechはMODIFY_AUDIO_SETTINGS権限を追加するなどして動作させることができた。全天球向けの認識精度向上など工夫の余地はまだ多くあると考えられる。
鍵となるTensorFlowInferenceInterfaceクラスは assetsの下にある学習済みモデル(Protocol Buffers(.pb)形式のファイル)をパラメータとして利用している。同様に配置して読み込むことで様々な学習モデルを試すことができる。

参考

https://codelabs.developers.google.com/codelabs/tensorflow-style-transfer-android/index.html

https://api.ricoh/docs/theta-plugin-reference/camera-api/

http://iti.hatenablog.jp/entry/2017/05/25/093328



@ mktshhr

Follow