#include "tensorflow/core/framework/op.h"

#include "tensorflow/core/framework/shape\_inference.h"

#include "tensorflow/core/framework/op\_kernel.h"

using namespace tensorflow;

REGISTER\_OP("VarLengthReader")

.Input("input1: int32")

.Output("output: int32")

.SetShapeFn([](::tensorflow::shape\_inference::InferenceContext\* c) {

::tensorflow::shape\_inference::ShapeHandle input;

// check that input has only 1 dimension.

TF\_RETURN\_IF\_ERROR(c->WithRank(c->input(0), 1, &input));

// there's no inference on output shape.

return Status::OK();

});

class VarLengthReaderOp : public OpKernel {

public:

explicit VarLengthReaderOp(OpKernelConstruction\* context) : OpKernel(context) {}

void Compute(OpKernelContext\* context) override {

// Grab the input tensor

const Tensor& input\_tensor = context->input(0);

auto input = input\_tensor.flat<int32>();

// get the first element in the input tensor, use it as output shape.

int32 len = input(0);

TensorShape output\_shape = {1, len};

// Create an output tensor, the size is determined by the content of input.

Tensor\* output\_tensor = nullptr;

OP\_REQUIRES\_OK(context, context->allocate\_output(0, output\_shape, &output\_tensor));

auto output\_flat = output\_tensor->flat<int32>();

// Fill output with ones.

const int N = output\_flat.size();

for (int i = 0; i < N; i++) {

output\_flat(i) = 1;

}

}

};

REGISTER\_KERNEL\_BUILDER(Name("VarLengthReader").Device(DEVICE\_CPU), VarLengthReaderOp);