NLP example project

This is a tutorial for setting up your project using TF-NLP library. Here we focus on the scaffolding of project and pay little attention to any modeling aspects.

Below we use classification as an example.

Setup your codebase

First you need to define the Task by inheirting it. Task is an abstraction of any machine learning task, here we focus on two things inputs and optimization target.

NOTE: We use BertClassifier as base model. You can shop other models here.

Step 1: build_inputs

Here we use CoLA, a binary classification task as an example.

TODO(saberkun): Add demo data instructions.

There are 4 fields we care about in the tf.Example, input_ids, input_mask, segment_ids and label_ids. Then we start with a simple data loader by inheriting the DataLoader interface.

```
class ClassificationDataLoader(data_loader.DataLoader):
  ...
  def _parse(self, record: Mapping[str, tf.Tensor]):
    """Parses raw tensors into a dict of tensors to be consumed by the model."""
    x = {
        'input_word_ids': record['input_ids'],
        'input_mask': record['input_mask'],
        'input_type_ids': record['segment_ids']
    }
    y = record['label_ids']
    return (x, y)
  ...
```

Overall, loader will translate the tf.Example to approiate format for model to consume. Then in Task.build_inputs, link the dataset like

```
def build_inputs(self):
  ...
  loader = classification_data_loader.ClassificationDataLoader(params)
  return loader.load(input_context)
```

Step 2: build_losses

We use standard cross entropy loss and make sure the `build_losses()` returns a float scalar Tensor.

```
def build_losses(self, labels, model_outputs, aux_losses=None):
  loss = tf.keras.losses.sparse_categorical_crossentropy(
      labels, tf.cast(model_outputs, tf.float32), from_logits=True)
  ...
```

Try the workflow locally.

We use a small BERT model for local trial and error. Below is the command:

```
# Assume you are under official/nlp/projects.

python3 example/train.py \
  --experiment=example_bert_classification_example \
  --config_file=example/local_example.yaml \
  --mode=train \
  --model_dir=/tmp/example_project_test/
```

The train binary translates the config file for the experiments. Usually you may just change the task import logics:

```
task_config = classification_example.ClassificationExampleConfig()
task = classification_example.ClassificationExampleTask(task_config)
```

TIPs: You can also check the unittest for better understanding.

Finetune

TF-NLP make it easy to start from a pretrained checkpoint, try below. This is done through configuring task.init_checkpoint in the YAML config below, see the base_task.initialize method for more details.

We use GCP TPU to demonstrate this.

```
EXP_NAME=bert_base_cola
EXP_TYPE=example_bert_classification_example
CONFIG_FILE=example/experiments/classification_ft_cola.yaml
TPU_NAME=experiment01
MODEL_DIR=your GCS bucket folder

python3 example/train.py \
  --experiment=$EXP_TYPE \
  --mode=train_and_eval \
  --tpu=$TPU_NAME \
  --model_dir=${MODEL_DIR}
  --config_file=${CONFIG_FILE}
```