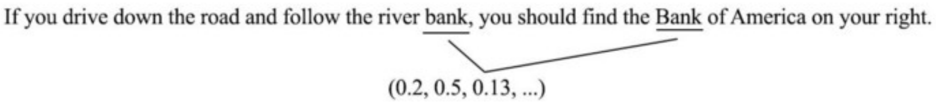
Exploring Bidirectional Encoder Representations from Transformers (BERT)

Table of Contents

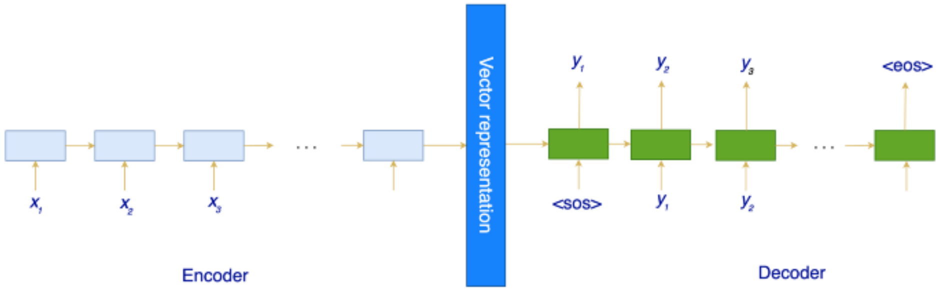
Introduction to BERT Architecture:

Transformer

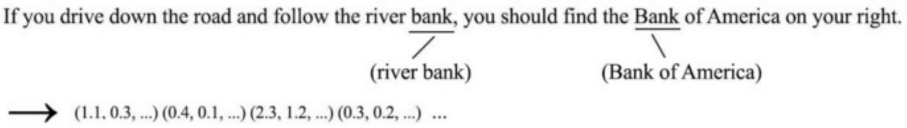
Many DL problems involve a major step of representing the input with a dense representation. This process forces the model to learn what is important in solving a problem. The extracted features are called the latent features, hidden variables, or a vector representation. Word embedding creates a vector representation of a word that we can manipulate with linear algebra. One major problem is words can have different meanings in different contexts. In the example below, word embedding uses the same vector in representing “bank”. But it has different meanings in the sentence.


If you drive down the road and follow the river bank, you should find the Bank of America on your right.
(0.2, 0.5, 0.13, ...)

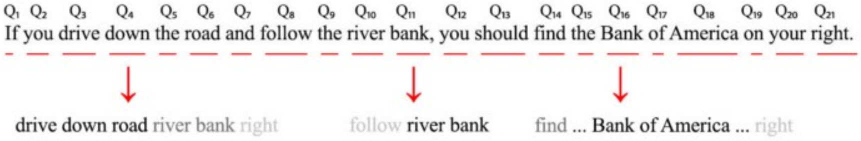
To create a dense representation of this sentence, we can apply RNN to parse a sequence of words in the form of embedding vectors. We gradually accumulate information in each timestep and produce a vector representation at the end of the pipeline. But one may argue that when the sentence is getting longer, early information may be forgotten or override. This may get worse if our input is a long paragraph.



Maybe, we should convert a sentence to a sequence of vectors instead, one vector per word. In addition, the context of a word will be considered during the encoding process through attention. For example, the word "bank" will be treated and encoded differently according to the context.


If you drive down the road and follow the river bank, you should find the Bank of America on your right.

Let's integrate this concept with attention using query, key, and value. We decompose sentences into single words. Each word acts as a value and we use the word itself as the key to its value.



Q_1 Q_2 Q_3 Q_4 Q_5 Q_6 Q_7 Q_8 Q_9 Q_{10} Q_{11} Q_{12} Q_{13} Q_{14} Q_{15} Q_{16} Q_{17} Q_{18} Q_{19} Q_{20} Q_{21}

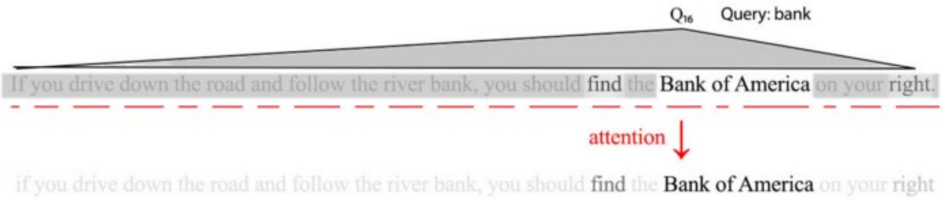
If you drive down the road and follow the river bank, you should find the Bank of America on your right.

drive down road river bank right

follow river bank

find ... Bank of America ... right

Each word form a single query. So the sentence above has 21 queries. How do we generate attention for a query, say Q_{16} for the word "bank"? We compute the relevancy of the query word "bank" with each key in the sentence. The attention is simply a weighted output of the values according to the relevancy. Conceptually, we "grey out" non-relevant values to form the attention.



If you drive down the road and follow the river bank, you should find the Bank of America on your right.

if you drive down the road and follow the river bank, you should find the Bank of America on your right

By going through Q_1 to Q_{21}, we collect all 21 attentions. This 21-vectors represent the sentence above.

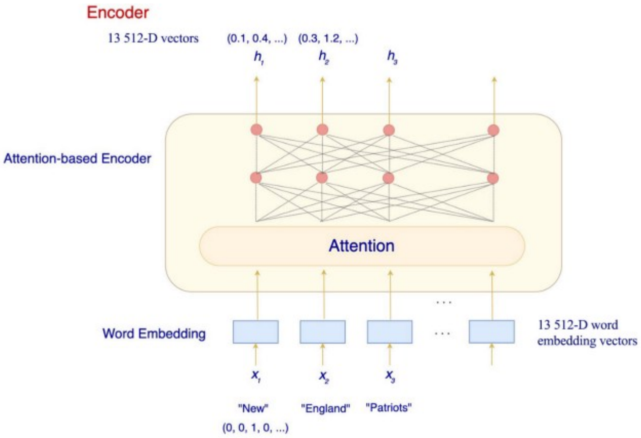
Transformer Encoder

We use the sentence below which contains 13 words.

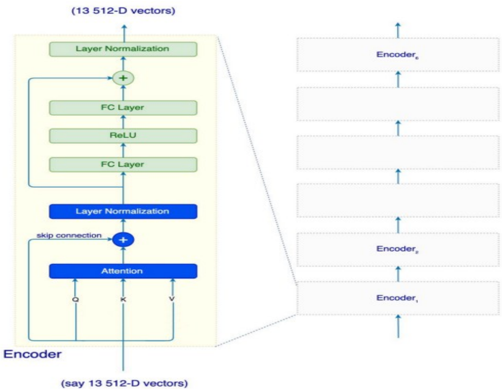
“New England Patriots win 14th straight regular-season game at home in Gillette stadium.”

In the encoding step, _Transformer uses learned word embedding_ to convert these 13 words, in one-hot-vector form, _into 13 512-D word embedding vectors_. Then they are passed into an attention-based encoder to pick the context information for each word.

For each word-embedding vector, there will be one output vector. These 13 word-embedding vectors will fit into position-wise fully connected layers (details later) to generate a sequence of 13 encoded vectors in representing the sentence. Each of these output vector h_i will be encoded in a 512-D vector. Conceptually, the output h_i encodes the word x_i with its context taking into consideration.



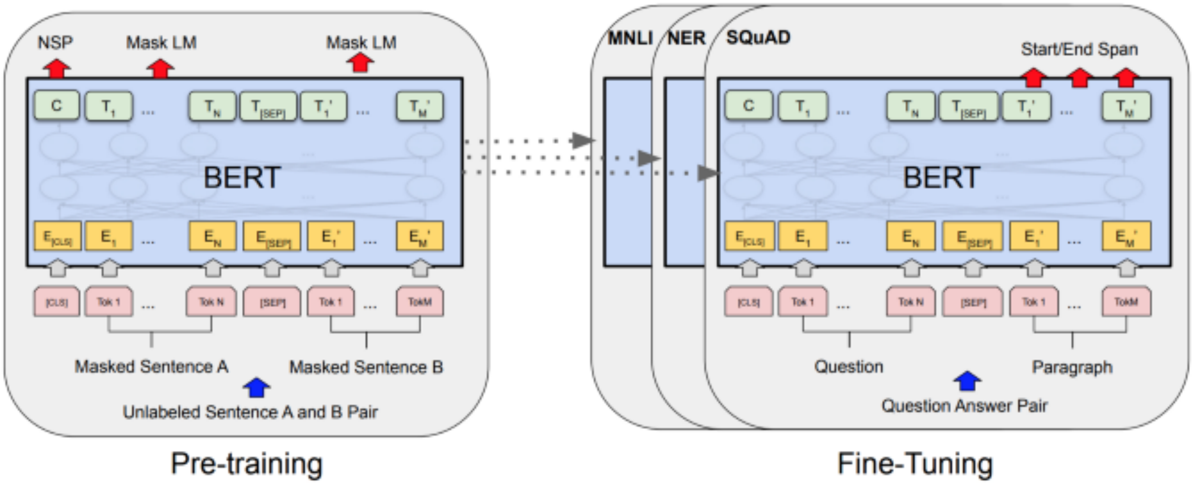
Let’s zoom into this attention-based encoder more. The encoder actually stacks up 6 encoders on the left below. The output of an encoder is fed to the encoder above. Each encoder takes 13 512-D vectors and output 13 512-D vectors. For the first decoder (encoder$_1$), the input is the 13 512-D word embedding vectors.



Strictly speaking, BERT is a training strategy, not a new architecture design

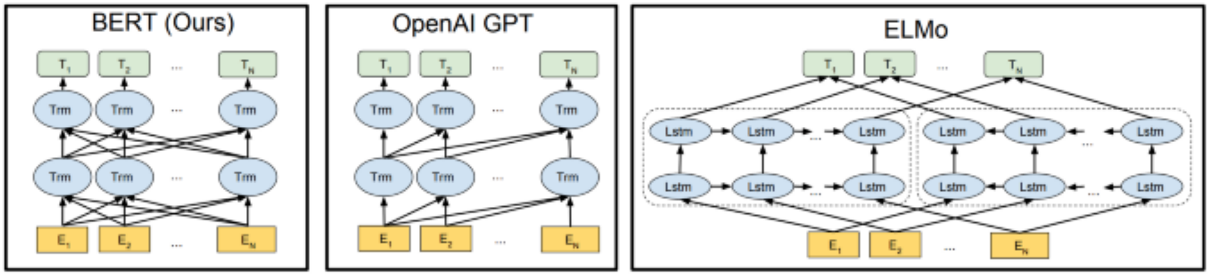
In BERT, a model is first pre-trained with data that requires no human labeling. Once it is done, the pre-trained model outputs a dense representation of the input. To solve other NLP tasks, like QA, we modify the model by simply adding a shallow DL layer connecting to the output of the original model. Then, we retrain the model with data and labels specific to the task.

In short, there is a pre-training phase in which we create a dense representation of the input (the left diagram below). The second phase returns the model with task-specific data, like MNLI or SQuAD, to solve the target NLP problem.



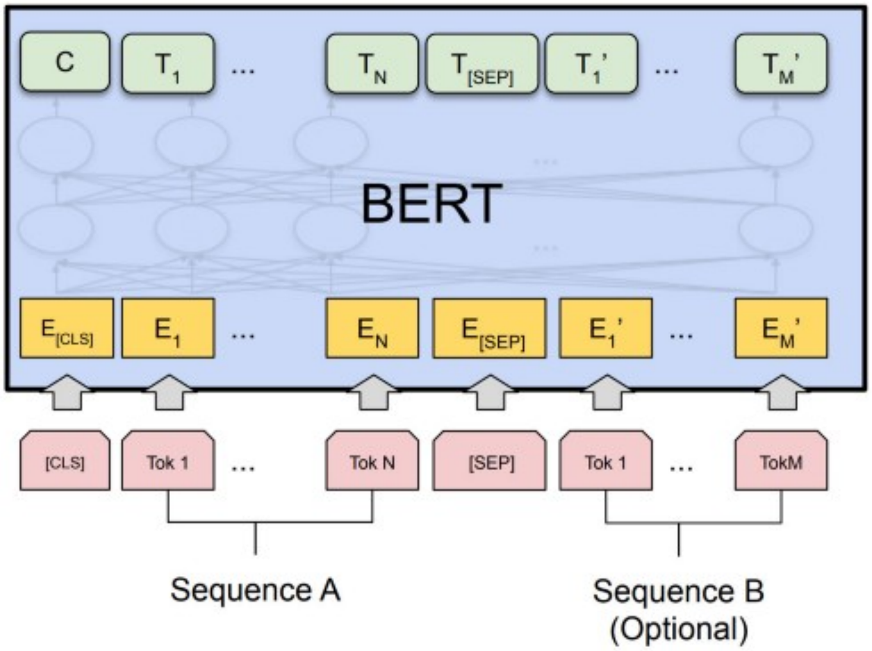
Model

BERT uses the Transformer encoder we discussed to create the vector representation. In contrast to other approaches, it discovers the context concurrent rather than directionally.



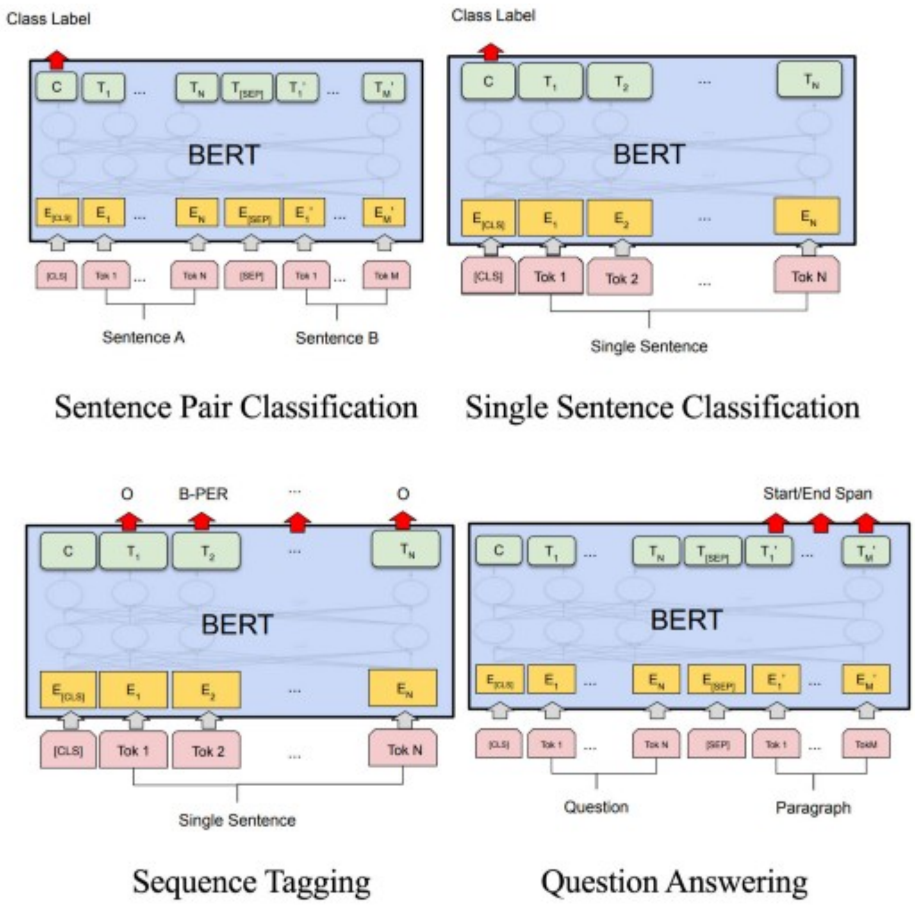
Input/Output Representations

But first, let’s define how input is assembled and what output is expected for the pre-trained model. First, the model needs to take one or two word-sequences to handle different spectrums of NLP tasks.



All input will start with a special token [CLS] (a special classification token). If the input composes of two sequences, a [SEP] token will put between Sequence A and Sequence B.

If the input has T tokens, including the added tokens, the output will have T outputs also. Different parts of the output will be used to make predictions for different NLP tasks. The first output is C (or sometimes written as the output [CLS] token). It is the only output used to derive a prediction for any NLP classification task. For non-classification tasks with only one sequence, we use the remaining outputs (without C). For QA, the outputs corresponding to the paragraph sequence will be used to derive the start and the end span of the answer.



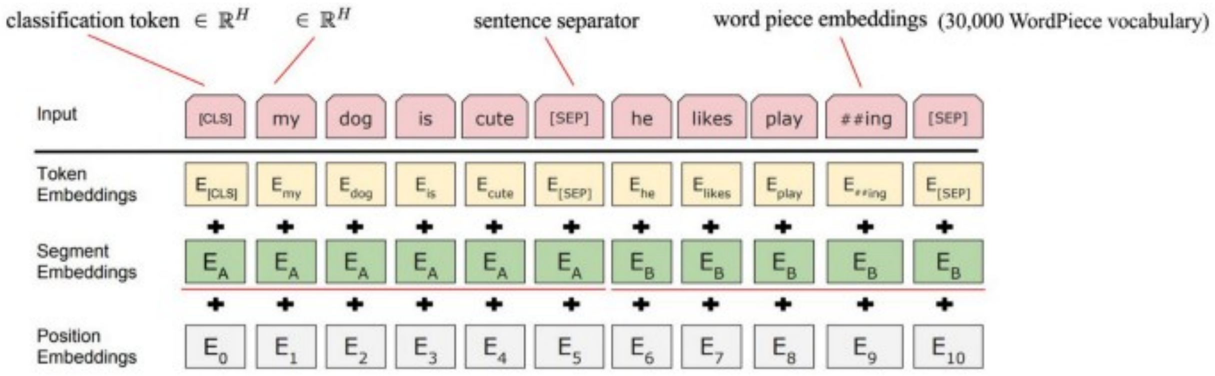
Sentence Pair Classification

Single Sentence Classification

Sequence Tagging

Question Answering

So, how do we compose the input embedding? In BERT, the input embedding composes of word piece embedding, segment embeddings, and position embedding of the same dimension. We add them together to form the final input embedding.

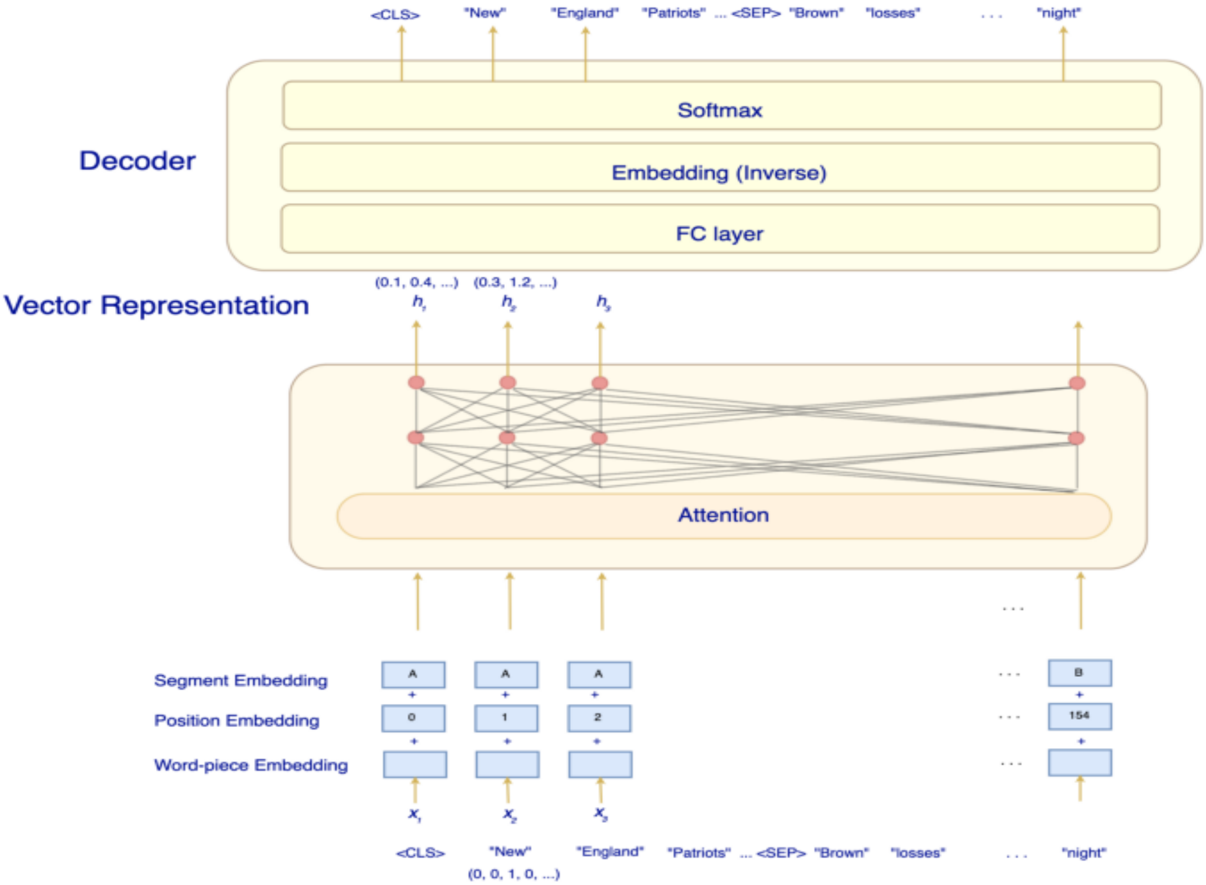

classification token $\in \mathbb{R}^H$
$\in \mathbb{R}^H$
sentence separator
word piece embeddings (30,000 WordPiece vocabulary)
Input
[CLS] my dog is cute [SEP] he likes play ##ing [SEP]
Token Embeddings
$E_{[CLS]}$ E_{my} E_{dog} E_{is} E_{cute} $E_{[SEP]}$ E_{he} E_{likes} E_{play} $E_{\#\#ing}$ $E_{[SEP]}$
Segment Embeddings
E_A E_A E_A E_A E_A E_A E_B E_B E_B E_B E_B
Position Embeddings
E_0 E_1 E_2 E_3 E_4 E_5 E_6 E_7 E_8 E_9 E_{10}

Instead of using every single word as tokens, BERT breaks a word into word pieces to reduce the vocabulary size (30,000 token vocabularies). For example, the word “helping” may decompose into “help” and “ing”. Then it applies an embedding matrix (V × H) to convert the one-hot vector R^V to R^H.

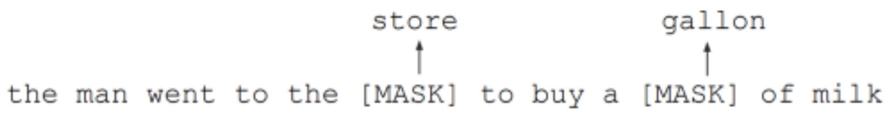
The segment embeddings model which sequence that tokens belong to. Does it belong to the first sentence or the second sentence. So it has a vocabulary size of two (segment A or B). Intuitively, it adds a constant offset to the embedding with value based on whether it belongs to sequence A or B. Mathematically, we apply an embedding matrix (2 × H) to convert R^2 to R^H. The last one is the position embedding in H-Dimension. It serves the same purpose in the Transformer in identifying the absolute or relative position of words.

Pretraining

BERT pre-trains the model using 2 NLP tasks. The first one is the Masked LM (Masked Language Model). As shown below, we use the Transformer decoder to generate a vector representation of the input. Then BERT applies a shallow deep decoder to reconstruct the word sequence(s) back.

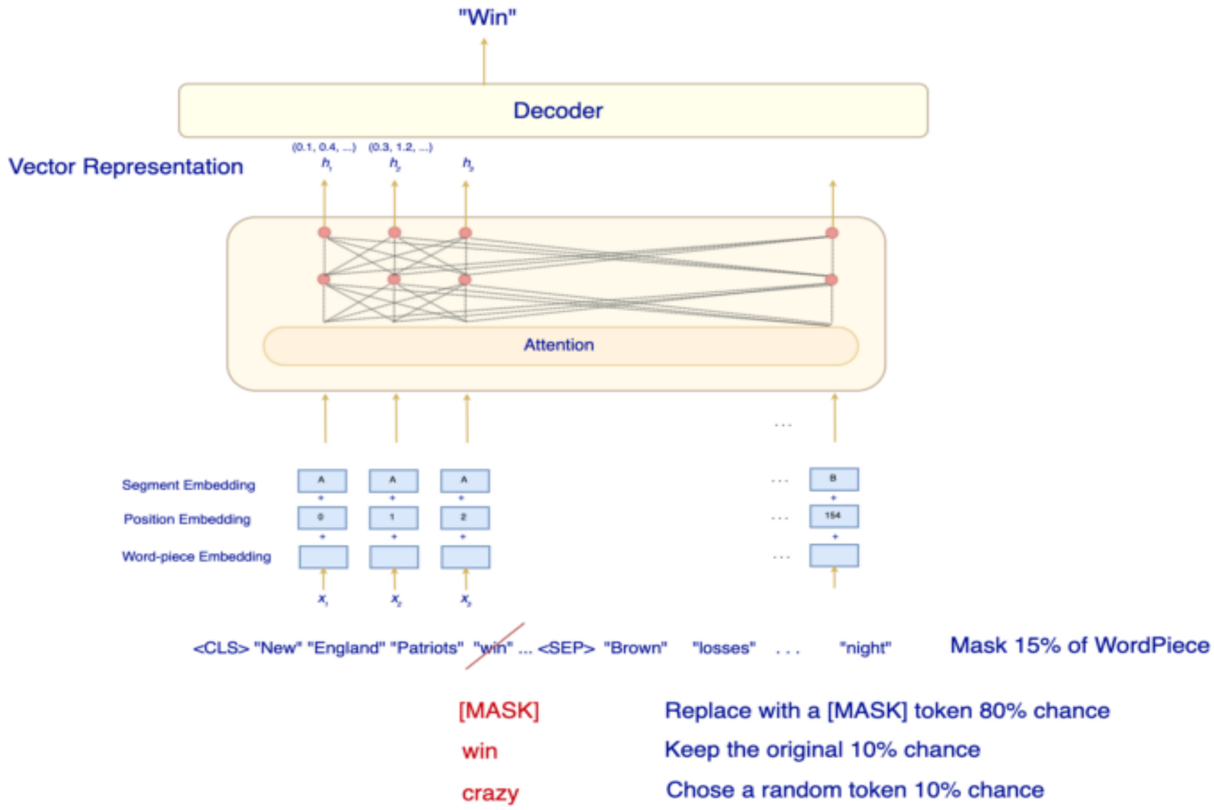


Here is an example of the Masked LM and BERT is trained to predict the missing words correctly.



Masked LM

In the Masked LM, BERT masks out 15% of the WordPiece. 80% of the masked WordPiece will be replaced with a [MASK] token, 10% with a random token and 10% will keep the original word. The loss is defined as how well BERT predicts the missing word, not the reconstruction error of the whole sequence.

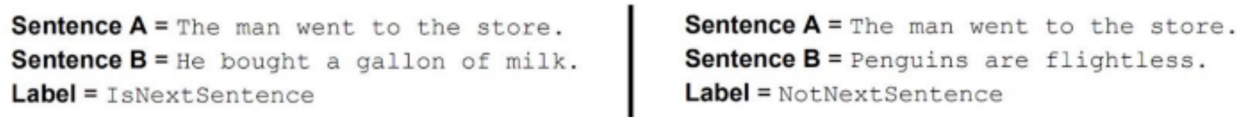


We do not replace 100% of the WordPiece with the [MASK] token. This teaches the model to predict missing words, not the final objective of creating vector representations for the sequences with context taken into consideration. BERT replaces 10% with random tokens and 10% with the original words. This encourages the model to learn what may be correct or what be wrong for the missing words.

Next Sentence Prediction (NSP)

The second pre-trained task is NSP. The key purpose is to create a representation in the output C that will encode the relations between Sequence A and B. To prepare the training input, in

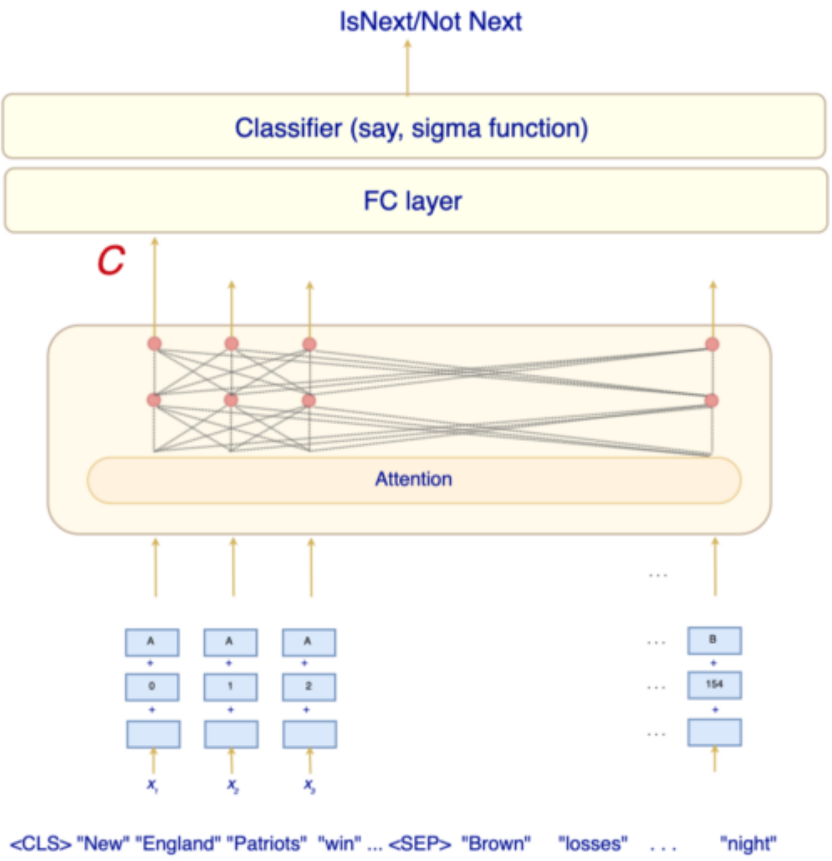
50% of the time, BERT uses two consecutive sentences as sequence A and B respectively. BERT expects the model to predict “IsNext”, i.e. sequence B should follow sequence A. For the remaining 50% of the time, BERT selects two-word sequences randomly and expect the prediction to be “Not Next”.



Sentence A = `The man went to the store.`
Sentence B = `He bought a gallon of milk.`
Label = `IsNextSentence`

Sentence A = `The man went to the store.`
Sentence B = `Penguins are flightless.`
Label = `NotNextSentence`

In this training, we take the output C and then classify it with a shallow classifier.



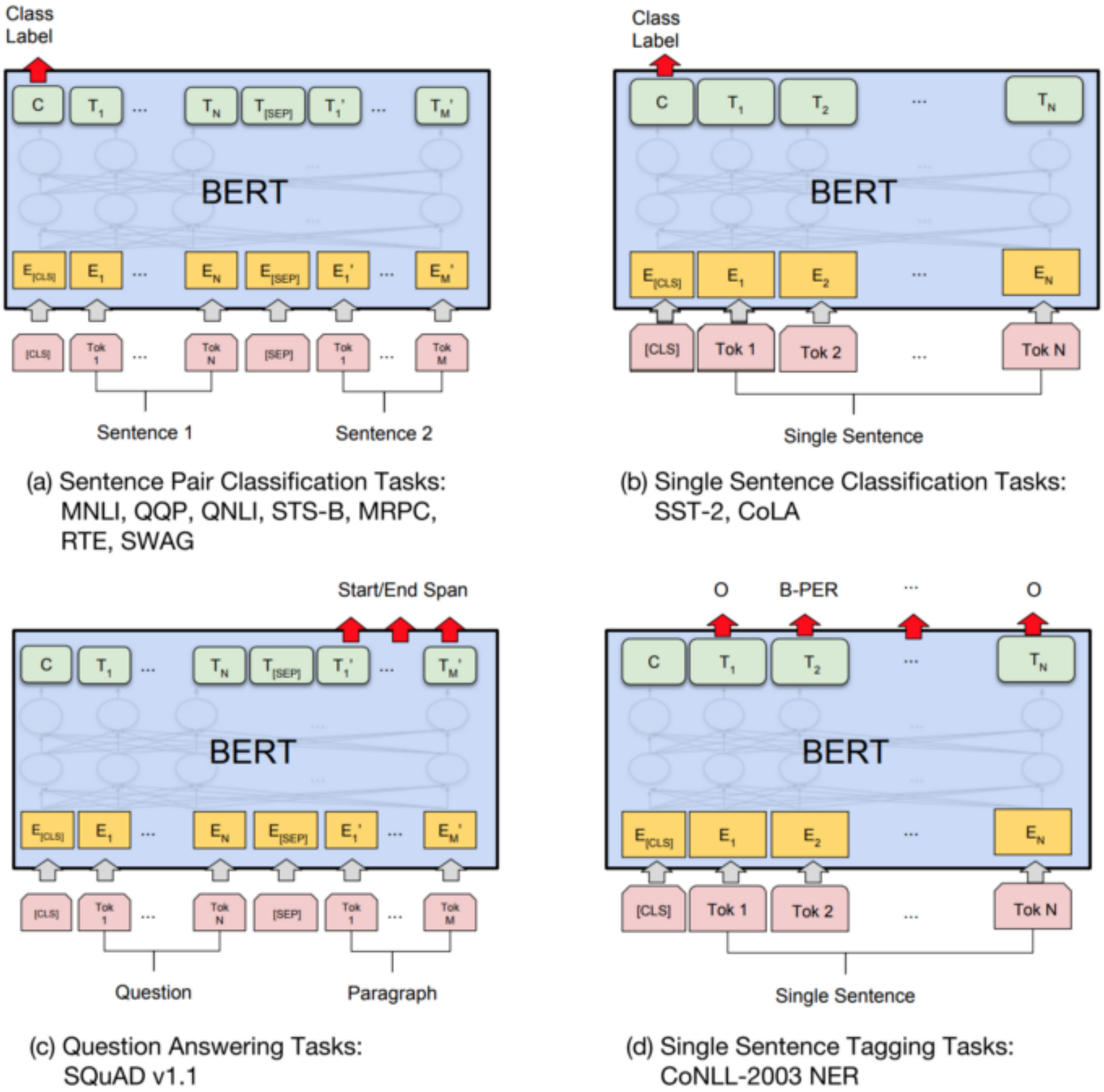
IsNext/Not Next
Classifier (say, sigma function)
FC layer
C
Attention
A
A
A
B
0
1
2
154
x_1
x_2
x_3
<CLS> "New" "England" "Patriots" "win" ... <SEP> "Brown" "losses" . . . "night"

<u>As noted, for both pre-training task, we create the training from a corpse without any human labeling.</u>

These two training tasks help BERT to train the vector representation of one or two word-sequences. Other than the context, it likely discovers other linguistics information including semantics and coreference.

Fine-tuning BERT

Once the model is pre-trained, we can add a shallow classifier for any NLP task or a decoder, similar to what we discussed in the pre-training step.



(a) Sentence Pair Classification Tasks: MNLI, QQP, QNLI, STS-B, MRPC, RTE, SWAG

(b) Single Sentence Classification Tasks: SST-2, CoLA

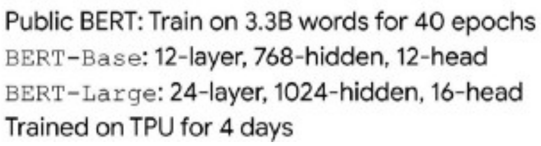
(c) Question Answering Tasks: SQuAD v1.1

(d) Single Sentence Tagging Tasks: CoNLL-2003 NER

Then, we fit the task-related data and the corresponding labels to refine all the model parameters end-to-end. That is how the model is trained and refined. So BERT is more on the training strategy rather than the model architecture. Its encoder is simply the Transformer encoder.

Model

But the model configuration in BERT is different from the Transformer paper. Here are a sample configuration used for the Transformer encoder in BERT.



Public BERT: Train on 3.3B words for 40 epochs
BERT-Base: 12-layer, 768-hidden, 12-head
BERT-Large: 24-layer, 1024-hidden, 16-head
Trained on TPU for 4 days

For example, the base model stacks up 12 decoders, instead of 6. Each output vector has a 768 dimension, and the attention uses 12 heads.

Source Code

For those interested in the source code for BERT, here is the source code from Google. For Transformer, here is the source code.

Sentence (and Sentence-Pair) classification tasks

Training a Classifier on Nvidia GTX 1060

Pre-Steps:

Download BERT-Base model `uncased_L-12_H-768_A-12 from following instructions in github.`

Training:

Step 1: Downloading GLUE dataset using script
Step 2: Download BERT-base model (pre-trained - uncased_L-12_H-768_A-12)
Step 3: Set below ENVs

```
export BERT_BASE_DIR=/path/to/bert/uncased_L-12_H-768_A-12
export GLUE_DIR=/path/to/glue
```

Step 4: Training Classifier

```
psakhamo@trainml:~/MyFolder/BERT_Exploration/bert$ python3 run_classifier.py --task_name=MRPC --do_train=true --do_eval=true --data_dir=$GLUE_DIR/MRPC --vocab_file=$BERT_BASE_DIR/vocab.txt --bert_config_file=$BERT_BASE_DIR/bert_config.json --init_checkpoint=$BERT_BASE_DIR/bert_model.ckpt --max_seq_length=64 --train_batch_size=12 --learning_rate=2e-5 --num_train_epochs=3.0 --output_dir=./out/
```

```
/home/psakhamo/.local/lib/python3.5/site-packages/tensorflow/python/framework/dtypes.py:516: FutureWarning: Passing (type, 1) or '1type' as a synonym of type is deprecated; in a future version of numpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_qint8 = np.dtype([("qint8", np.int8, 1)])
/home/psakhamo/.local/lib/python3.5/site-packages/tensorflow/python/framework/dtypes.py:517: FutureWarning: Passing (type, 1) or '1type' as a synonym of type is deprecated; in a future version of numpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_quint8 = np.dtype([("quint8", np.uint8, 1)])
/home/psakhamo/.local/lib/python3.5/site-packages/tensorflow/python/framework/dtypes.py:518: FutureWarning: Passing (type, 1) or '1type' as a synonym of type is deprecated; in a future version of numpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_qint16 = np.dtype([("qint16", np.int16, 1)])
/home/psakhamo/.local/lib/python3.5/site-packages/tensorflow/python/framework/dtypes.py:519: FutureWarning: Passing (type, 1) or '1type' as a synonym of type is deprecated; in a future version of numpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_quint16 = np.dtype([("quint16", np.uint16, 1)])
/home/psakhamo/.local/lib/python3.5/site-packages/tensorflow/python/framework/dtypes.py:520: FutureWarning: Passing (type, 1) or '1type' as a synonym of type is deprecated; in a future version of numpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_qint32 = np.dtype([("qint32", np.int32, 1)])
/home/psakhamo/.local/lib/python3.5/site-packages/tensorflow/python/framework/dtypes.py:525: FutureWarning: Passing (type, 1) or '1type' as a synonym of type is deprecated; in a future version of numpy, it will be understood as (type, (1,)) / '(1,)type'.
  np_resource = np.dtype([("resource", np.ubyte, 1)])
/home/psakhamo/.local/lib/python3.5/site-packages/tensorboard/compat/tensorflow_stub/dtypes.py:541: FutureWarning: Passing (type, 1) or '1type' as a synonym of type is deprecated; in a future version of numpy, it will be understood as (type, (1,)) / '(1,)type'.
```

```
  _np_qint8 = np.dtype([("qint8", np.int8, 1)])
/home/psakhamo/.local/lib/python3.5/site-packages/tensorboard/compat/tensorflow_stub/dtypes.py:542: FutureWarning: Passing (type, 1) or '1type' as a synonym of type is deprecated; in a future version of numpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_quint8 = np.dtype([("quint8", np.uint8, 1)])
/home/psakhamo/.local/lib/python3.5/site-packages/tensorboard/compat/tensorflow_stub/dtypes.py:543: FutureWarning: Passing (type, 1) or '1type' as a synonym of type is deprecated; in a future version of numpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_qint16 = np.dtype([("qint16", np.int16, 1)])
/home/psakhamo/.local/lib/python3.5/site-packages/tensorboard/compat/tensorflow_stub/dtypes.py:544: FutureWarning: Passing (type, 1) or '1type' as a synonym of type is deprecated; in a future version of numpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_quint16 = np.dtype([("quint16", np.uint16, 1)])
/home/psakhamo/.local/lib/python3.5/site-packages/tensorboard/compat/tensorflow_stub/dtypes.py:545: FutureWarning: Passing (type, 1) or '1type' as a synonym of type is deprecated; in a future version of numpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_qint32 = np.dtype([("qint32", np.int32, 1)])
/home/psakhamo/.local/lib/python3.5/site-packages/tensorboard/compat/tensorflow_stub/dtypes.py:550: FutureWarning: Passing (type, 1) or '1type' as a synonym of type is deprecated; in a future version of numpy, it will be understood as (type, (1,)) / '(1,)type'.
  np_resource = np.dtype([("resource", np.ubyte, 1)])
WARNING: Logging before flag parsing goes to stderr.
W1104 00:15:28.193943 139815656863488 deprecation_wrapper.py:119] From /home/psakhamo/MyFolder/BERT_Exploration/bert/optimization.py:87: The name tf.train.Optimizer is deprecated. Please use tf.compat.v1.train.Optimizer instead.

W1104 00:15:28.195036 139815656863488 deprecation_wrapper.py:119] From run_classifier.py:981: The name tf.app.run is deprecated. Please use tf.compat.v1.app.run instead.

W1104 00:15:28.195385 139815656863488 deprecation_wrapper.py:119] From run_classifier.py:784: The name tf.logging.set_verbosity is deprecated. Please use tf.compat.v1.logging.set_verbosity instead.

W1104 00:15:28.195476 139815656863488 deprecation_wrapper.py:119] From run_classifier.py:784: The name tf.logging.INFO is deprecated. Please use tf.compat.v1.logging.INFO instead.

W1104 00:15:28.195734 139815656863488 deprecation_wrapper.py:119] From /home/psakhamo/MyFolder/BERT_Exploration/bert/modeling.py:93: The name tf.gfile.GFile is deprecated. Please use tf.io.gfile.GFile instead.

W1104 00:15:28.196100 139815656863488 deprecation_wrapper.py:119] From run_classifier.py:808: The name tf.gfile.MakeDirs is deprecated. Please use tf.io.gfile.makedirs instead.

W1104 00:15:28.526511 139815656863488 lazy_loader.py:50]
The TensorFlow contrib module will not be included in TensorFlow 2.0.
For more information, please see:
  * https://github.com/tensorflow/community/blob/master/rfcs/20180907-contrib-sunset.md
  * https://github.com/tensorflow/addons
  * https://github.com/tensorflow/io (for I/O related ops)
If you depend on functionality not listed there, please file an issue.

W1104 00:15:28.526825 139815656863488 deprecation_wrapper.py:119] From run_classifier.py:199: The name tf.gfile.Open is deprecated. Please use tf.io.gfile.GFile instead.

W1104 00:15:28.552463 139815656863488 estimator.py:1984] Estimator's model_fn (<function model_fn_builder.<locals>.model_fn at 0x7f28fab848c8>) includes params argument, but params are not passed to Estimator.
I1104 00:15:28.553177 139815656863488 estimator.py:209] Using config: {'_tpu_config': TPUConfig(iterations_per_loop=1000, num_shards=8, num_cores_per_replica=None, per_host_input_for_training=3, tpu_job_name=None, initial_infeed_sleep_secs=None, input_partition_dims=None, eval_training_input_configuration=2), '_experimental_max_worker_delay_secs': None, '_tf_random_seed': None, '_num_ps_replicas': 0, '_num_worker_replicas': 1, '_session_config': allow_soft_placement: true
graph_options {
  rewrite_options {
    meta_optimizer_iterations: ONE
  }
}
, '_cluster_spec': <tensorflow.python.training.server_lib.ClusterSpec object at 0x7f28fb0e8c18>, '_eval_distribute': None, '_save_summary_steps': 100, '_save_checkpoints_steps': 1000, '_keep_checkpoint_max': 5, '_experimental_distribute': None, '_is_chief': True, '_master': '', '_cluster': None, '_protocol': None, '_model_dir': './out/', '_keep_checkpoint_every_n_hours': 10000, '_service': None, '_task_id': 0, '_evaluation_master': '', '_global_id_in_cluster': 0, '_train_distribute': None, '_save_checkpoints_secs': None, '_device_fn': None, '_task_type': 'worker', '_log_step_count_steps': None}
I1104 00:15:28.553448 139815656863488 tpu_context.py:209] _TPUContext: eval_on_tpu True
W1104 00:15:28.553603 139815656863488 tpu_context.py:211] eval_on_tpu ignored because use_tpu is False.
W1104 00:15:28.553706 139815656863488 deprecation_wrapper.py:119] From run_classifier.py:483: The name tf.python_io.TFRecordWriter is deprecated. Please use tf.io.TFRecordWriter instead.

W1104 00:15:28.554040 139815656863488 deprecation_wrapper.py:119] From run_classifier.py:487: The name tf.logging.info is deprecated. Please use tf.compat.v1.logging.info instead.

I1104 00:15:28.554097 139815656863488 run_classifier.py:487] Writing example 0 of 3668
I1104 00:15:28.554683 139815656863488 run_classifier.py:461] *** Example ***
I1104 00:15:28.554750 139815656863488 run_classifier.py:462] guid: train-1
I1104 00:15:28.554806 139815656863488 run_classifier.py:464] tokens: [CLS] am ##ro ##zi accused his brother , whom he called " the witness " , of deliberately di ##stor ##ting his evidence . [SEP] referring to him as only " the witness " , am ##ro ##zi accused his brother of deliberately di ##stor ##ting his evidence . [SEP]
I1104 00:15:28.554861 139815656863488 run_classifier.py:465] input_ids: 101 2572 3217 5831 5496 2010 2567 1010 3183 2002 2170 1000 1996 7409 1000 1010 1997 9969 4487 23809 3436 2010 3350 1012 102 7727 2000 2032 2004 2069 1000 1996 7409 1000 1010 2572 3217 5831 5496 2010 2567 1997 9969 4487 23809 3436 2010 3350 1012 102 0 0 0 0 0 0 0 0 0 0 0 0 0 0
I1104 00:15:28.554913 139815656863488 run_classifier.py:466] input_mask: 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0
0
I1104 00:15:28.554962 139815656863488 run_classifier.py:467] segment_ids: 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0
0 0
I1104 00:15:28.555003 139815656863488 run_classifier.py:468] label: 1 (id = 1)
I1104 00:15:28.555764 139815656863488 run_classifier.py:461] *** Example ***
I1104 00:15:28.555824 139815656863488 run_classifier.py:462] guid: train-2
I1104 00:15:28.555896 139815656863488 run_classifier.py:464] tokens: [CLS] yu ##ca ##ip ##a owned dominic ##k ' s before selling the chain to safe ##way in 1998 for $ 2 . 5 billion . [SEP] yu ##ca ##ip ##a bought dominic ##k ' s in 1995 for $ 69 ##3 million and sold it to safe ##way for $ 1 . 8 billion in 1998 . [SEP]
I1104 00:15:28.555967 139815656863488 run_classifier.py:465] input_ids: 101 9805 3540 11514 2050 3079 11282 2243 1005 1055 2077 4855 1996 4677 2000 3647 4576 1999 2687 2005 1002 1016 1012 1019 4551 1012 102 9805 3540 11514 2050 4149 11282 2243 1005 1055 1999 2786 2005 1002 6353 2509 2454 1998 2853 2009 2000 3647 4576 2005 1002 1015 1012 1022 4551 1999 2687 1012 102 0 0 0 0 0
I1104 00:15:28.556046 139815656863488 run_classifier.py:466] input_mask: 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0
0
I1104 00:15:28.556125 139815656863488 run_classifier.py:467] segment_ids: 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0
0 0
I1104 00:15:28.556179 139815656863488 run_classifier.py:468] label: 0 (id = 0)
I1104 00:15:28.556881 139815656863488 run_classifier.py:461] *** Example ***
```

```
I1104 00:15:28.556952 139815656863488 run_classifier.py:462] guid: train-3
I1104 00:15:28.557036 139815656863488 run_classifier.py:464] tokens: [CLS] they had published an advertisement on the internet on june 10 , offering the cargo for sale , he added . [SEP] on june 10 , the ship ' s owners had published an advertisement on the internet , offering the explosives for sale . [SEP]
I1104 00:15:28.557091 139815656863488 run_classifier.py:465] input_ids: 101 2027 2018 2405 2019 15147 2006 1996 4274 2006 2238 2184 1010 5378 1996 6636 2005 5096 1010 2002 2794 1012 102 2006 2238 2184 1010 1996 2911 1005 1055 5608 2018 2405 2019 15147 2006 1996 4274 1010 5378 1996 14792 2005 5096 1012 102 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
I1104 00:15:28.557143 139815656863488 run_classifier.py:466] input_mask: 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
I1104 00:15:28.557194 139815656863488 run_classifier.py:467] segment_ids: 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
I1104 00:15:28.557235 139815656863488 run_classifier.py:468] label: 1 (id = 1)
I1104 00:15:28.557916 139815656863488 run_classifier.py:461] *** Example ***
I1104 00:15:28.557972 139815656863488 run_classifier.py:462] guid: train-4
I1104 00:15:28.558031 139815656863488 run_classifier.py:464] tokens: [CLS] around 03 ##35 gm ##t , tab shares were up 19 cents , or 4 . 4 % , at a $ 4 . 56 , having earlier set a record high of a $ 4 [SEP] tab shares jumped 20 cents , or 4 . 6 % , to set a record closing high at a $ 4 . 57 . [SEP]
I1104 00:15:28.558086 139815656863488 run_classifier.py:465] input_ids: 101 2105 6021 19481 13938 2102 1010 21628 6661 2020 2039 2539 16653 1010 2030 1018 1012 1018 1003 1010 2012 1037 1002 1018 1012 5179 1010 2383 3041 2275 1037 2501 2152 1997 1037 1002 1018 102 21628 6661 5598 2322 16653 1010 2030 1018 1012 1020 1003 1010 2000 2275 1037 2501 5494 2152 2012 1037 1002 1018 1012 5401 1012 102
I1104 00:15:28.558138 139815656863488 run_classifier.py:466] input_mask: 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
I1104 00:15:28.558188 139815656863488 run_classifier.py:467] segment_ids: 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
I1104 00:15:28.558230 139815656863488 run_classifier.py:468] label: 0 (id = 0)
I1104 00:15:28.558902 139815656863488 run_classifier.py:461] *** Example ***
I1104 00:15:28.558958 139815656863488 run_classifier.py:462] guid: train-5
I1104 00:15:28.559017 139815656863488 run_classifier.py:464] tokens: [CLS] the stock rose $ 2 . 11 , or about 11 percent , to close friday at $ 21 . 51 on the new york stock exchange . [SEP] pg & e corp . shares jumped $ 1 . 63 or 8 percent to $ 21 . 03 on the new york stock exchange on friday . [SEP]
I1104 00:15:28.559072 139815656863488 run_classifier.py:465] input_ids: 101 1996 4518 3123 1002 1016 1012 2340 1010 2030 2055 2340 3867 1010 2000 2485 5958 2012 1002 2538 1012 4868 2006 1996 2047 2259 4518 3863 1012 102 18720 1004 1041 13058 1012 6661 5598 1002 1015 1012 6191 2030 1022 3867 2000 1002 2538 1012 6021 2006 1996 2047 2259 4518 3863 2006 5958 1012 102 0 0 0 0 0
I1104 00:15:28.559125 139815656863488 run_classifier.py:466] input_mask: 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0
I1104 00:15:28.559176 139815656863488 run_classifier.py:467] segment_ids: 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0
I1104 00:15:28.559216 139815656863488 run_classifier.py:468] label: 1 (id = 1)
I1104 00:15:30.872557 139815656863488 run_classifier.py:871] ***** Running training *****
I1104 00:15:30.872689 139815656863488 run_classifier.py:872]   Num examples = 3668
I1104 00:15:30.872946 139815656863488 run_classifier.py:873]   Batch size = 12
I1104 00:15:30.873022 139815656863488 run_classifier.py:874]   Num steps = 917
W1104 00:15:30.873129 139815656863488 deprecation_wrapper.py:119] From run_classifier.py:514: The name tf.FixedLenFeature is deprecated. Please use tf.io.FixedLenFeature instead.

W1104 00:15:30.879420 139815656863488 deprecation.py:323] From /home/psakhamo/.local/lib/python3.5/site-packages/tensorflow/python/training/training_util.py:236: Variable.initialized_value (from tensorflow.python.ops.variables) is deprecated and will be removed in a future version.
Instructions for updating:
Use Variable.read_value. Variables in 2.X are initialized automatically both in eager and graph (inside tf.defun) contexts.
W1104 00:15:30.899166 139815656863488 deprecation.py:323] From run_classifier.py:550: map_and_batch (from tensorflow.contrib.data.python.ops.batching) is deprecated and will be removed in a future version.
Instructions for updating:
Use `tf.data.experimental.map_and_batch(...)`.
W1104 00:15:30.899319 139815656863488 deprecation.py:323] From /home/psakhamo/.local/lib/python3.5/site-packages/tensorflow/contrib/data/python/ops/batching.py:273: map_and_batch (from tensorflow.python.data.experimental.ops.batching) is deprecated and will be removed in a future version.
Instructions for updating:
Use `tf.data.Dataset.map(map_func, num_parallel_calls)` followed by `tf.data.Dataset.batch(batch_size, drop_remainder)`. Static tf.data optimizations will take care of using the fused implementation.
W1104 00:15:30.900217 139815656863488 deprecation_wrapper.py:119] From run_classifier.py:523: The name tf.parse_single_example is deprecated. Please use tf.io.parse_single_example instead.

W1104 00:15:30.903018 139815656863488 deprecation.py:323] From run_classifier.py:530: to_int32 (from tensorflow.python.ops.math_ops) is deprecated and will be removed in a future version.
Instructions for updating:
Use `tf.cast` instead.
I1104 00:15:30.915224 139815656863488 estimator.py:1145] Calling model_fn.
I1104 00:15:30.915369 139815656863488 tpu_estimator.py:2965] Running train on CPU
I1104 00:15:30.915614 139815656863488 run_classifier.py:627] *** Features ***
I1104 00:15:30.915710 139815656863488 run_classifier.py:629]   name = input_ids, shape = (12, 64)
I1104 00:15:30.915802 139815656863488 run_classifier.py:629]   name = input_mask, shape = (12, 64)
I1104 00:15:30.915860 139815656863488 run_classifier.py:629]   name = is_real_example, shape = (12,)
I1104 00:15:30.915916 139815656863488 run_classifier.py:629]   name = label_ids, shape = (12,)
I1104 00:15:30.915974 139815656863488 run_classifier.py:629]   name = segment_ids, shape = (12, 64)
W1104 00:15:30.916516 139815656863488 deprecation_wrapper.py:119] From /home/psakhamo/MyFolder/BERT_Exploration/bert/modeling.py:171: The name tf.variable_scope is deprecated. Please use tf.compat.v1.variable_scope instead.

W1104 00:15:30.917510 139815656863488 deprecation_wrapper.py:119] From /home/psakhamo/MyFolder/BERT_Exploration/bert/modeling.py:409: The name tf.get_variable is deprecated. Please use tf.compat.v1.get_variable instead.

W1104 00:15:30.932637 139815656863488 deprecation_wrapper.py:119] From /home/psakhamo/MyFolder/BERT_Exploration/bert/modeling.py:490: The name tf.assert_less_equal is deprecated. Please use tf.compat.v1.assert_less_equal instead.

W1104 00:15:30.953652 139815656863488 deprecation.py:506] From /home/psakhamo/MyFolder/BERT_Exploration/bert/modeling.py:358: calling dropout (from tensorflow.python.ops.nn_ops) with keep_prob is deprecated and will be removed in a future version.
Instructions for updating:
Please use `rate` instead of `keep_prob`. Rate should be set to `rate = 1 - keep_prob`.
W1104 00:15:30.962591 139815656863488 deprecation.py:323] From /home/psakhamo/MyFolder/BERT_Exploration/bert/modeling.py:671: dense (from tensorflow.python.layers.core) is deprecated and will be removed in a future version.
Instructions for updating:
Use keras.layers.dense instead.
W1104 00:15:32.592846 139815656863488 deprecation_wrapper.py:119] From run_classifier.py:661: The name tf.train.init_from_checkpoint is deprecated. Please use tf.compat.v1.train.init_from_checkpoint instead.
```

```
I1104 00:15:33.033515 139815656863488 run_classifier.py:663] **** Trainable Variables ****
I1104 00:15:33.033651 139815656863488 run_classifier.py:669]   name = bert/embeddings/word_embeddings:0, shape = (30522, 768), *INIT_FROM_CKPT*
I1104 00:15:33.033735 139815656863488 run_classifier.py:669]   name = bert/embeddings/token_type_embeddings:0, shape = (2, 768), *INIT_FROM_CKPT*
I1104 00:15:33.033793 139815656863488 run_classifier.py:669]   name = bert/embeddings/position_embeddings:0, shape = (512, 768), *INIT_FROM_CKPT*
I1104 00:15:33.033846 139815656863488 run_classifier.py:669]   name = bert/embeddings/LayerNorm/beta:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:15:33.033895 139815656863488 run_classifier.py:669]   name = bert/embeddings/LayerNorm/gamma:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:15:33.033943 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_0/attention/self/query/kernel:0, shape = (768, 768), *INIT_FROM_CKPT*
I1104 00:15:33.033992 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_0/attention/self/query/bias:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:15:33.034040 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_0/attention/self/key/kernel:0, shape = (768, 768), *INIT_FROM_CKPT*
I1104 00:15:33.034089 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_0/attention/self/key/bias:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:15:33.034136 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_0/attention/self/value/kernel:0, shape = (768, 768), *INIT_FROM_CKPT*
I1104 00:15:33.034185 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_0/attention/self/value/bias:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:15:33.034232 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_0/attention/output/dense/kernel:0, shape = (768, 768), *INIT_FROM_CKPT*
I1104 00:15:33.034280 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_0/attention/output/dense/bias:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:15:33.034327 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_0/attention/output/LayerNorm/beta:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:15:33.034373 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_0/attention/output/LayerNorm/gamma:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:15:33.034419 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_0/intermediate/dense/kernel:0, shape = (768, 3072), *INIT_FROM_CKPT*
I1104 00:15:33.034467 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_0/intermediate/dense/bias:0, shape = (3072,), *INIT_FROM_CKPT*
I1104 00:15:33.034513 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_0/output/dense/kernel:0, shape = (3072, 768), *INIT_FROM_CKPT*
I1104 00:15:33.034562 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_0/output/dense/bias:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:15:33.034609 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_0/output/LayerNorm/beta:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:15:33.034655 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_0/output/LayerNorm/gamma:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:15:33.034700 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_1/attention/self/query/kernel:0, shape = (768, 768), *INIT_FROM_CKPT*
I1104 00:15:33.034749 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_1/attention/self/query/bias:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:15:33.034795 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_1/attention/self/key/kernel:0, shape = (768, 768), *INIT_FROM_CKPT*
I1104 00:15:33.034844 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_1/attention/self/key/bias:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:15:33.034890 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_1/attention/self/value/kernel:0, shape = (768, 768), *INIT_FROM_CKPT*
…..
…..
I1104 00:15:33.038888 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_6/attention/output/LayerNorm/gamma:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:15:33.038933 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_6/intermediate/dense/kernel:0, shape = (768, 3072), *INIT_FROM_CKPT*
I1104 00:15:33.038982 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_6/intermediate/dense/bias:0, shape = (3072,), *INIT_FROM_CKPT*
I1104 00:15:33.039028 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_6/output/dense/kernel:0, shape = (3072, 768), *INIT_FROM_CKPT*
I1104 00:15:33.039076 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_6/output/dense/bias:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:15:33.039122 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_6/output/LayerNorm/beta:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:15:33.039168 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_6/output/LayerNorm/gamma:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:15:33.039214 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_7/attention/self/query/kernel:0, shape = (768, 768), *INIT_FROM_CKPT*
I1104 00:15:33.039262 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_7/attention/self/query/bias:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:15:33.039309 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_7/attention/self/key/kernel:0, shape = (768, 768), *INIT_FROM_CKPT*
I1104 00:15:33.039357 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_7/attention/self/key/bias:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:15:33.039402 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_7/attention/self/value/kernel:0, shape = (768, 768), *INIT_FROM_CKPT*
I1104 00:15:33.039450 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_7/attention/self/value/bias:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:15:33.039496 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_7/attention/output/dense/kernel:0, shape = (768, 768), *INIT_FROM_CKPT*
I1104 00:15:33.039544 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_7/attention/output/dense/bias:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:15:33.039590 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_7/attention/output/LayerNorm/beta:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:15:33.039636 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_7/attention/output/LayerNorm/gamma:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:15:33.039682 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_7/intermediate/dense/kernel:0, shape = (768, 3072), *INIT_FROM_CKPT*
I1104 00:15:33.039730 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_7/intermediate/dense/bias:0, shape = (3072,), *INIT_FROM_CKPT*
I1104 00:15:33.039776 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_7/output/dense/kernel:0, shape = (3072, 768), *INIT_FROM_CKPT*
I1104 00:15:33.039824 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_7/output/dense/bias:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:15:33.039870 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_7/output/LayerNorm/beta:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:15:33.039915 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_7/output/LayerNorm/gamma:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:15:33.039960 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_8/attention/self/query/kernel:0, shape = (768, 768), *INIT_FROM_CKPT*
I1104 00:15:33.040008 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_8/attention/self/query/bias:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:15:33.040054 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_8/attention/self/key/kernel:0, shape = (768, 768), *INIT_FROM_CKPT*
I1104 00:15:33.040101 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_8/attention/self/key/bias:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:15:33.040147 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_8/attention/self/value/kernel:0, shape = (768, 768), *INIT_FROM_CKPT*
I1104 00:15:33.040195 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_8/attention/self/value/bias:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:15:33.040241 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_8/attention/output/dense/kernel:0, shape = (768, 768), *INIT_FROM_CKPT*
I1104 00:15:33.040289 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_8/attention/output/dense/bias:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:15:33.040335 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_8/attention/output/LayerNorm/beta:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:15:33.040380 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_8/attention/output/LayerNorm/gamma:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:15:33.040426 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_8/intermediate/dense/kernel:0, shape = (768, 3072), *INIT_FROM_CKPT*
I1104 00:15:33.040474 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_8/intermediate/dense/bias:0, shape = (3072,), *INIT_FROM_CKPT*
I1104 00:15:33.040523 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_8/output/dense/kernel:0, shape = (3072, 768), *INIT_FROM_CKPT*
I1104 00:15:33.040573 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_8/output/dense/bias:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:15:33.040619 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_8/output/LayerNorm/beta:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:15:33.040664 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_8/output/LayerNorm/gamma:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:15:33.040710 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_9/attention/self/query/kernel:0, shape = (768, 768), *INIT_FROM_CKPT*
I1104 00:15:33.040758 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_9/attention/self/query/bias:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:15:33.040803 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_9/attention/self/key/kernel:0, shape = (768, 768), *INIT_FROM_CKPT*
I1104 00:15:33.040851 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_9/attention/self/key/bias:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:15:33.040896 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_9/attention/self/value/kernel:0, shape = (768, 768), *INIT_FROM_CKPT*
I1104 00:15:33.040943 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_9/attention/self/value/bias:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:15:33.040989 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_9/attention/output/dense/kernel:0, shape = (768, 768), *INIT_FROM_CKPT*
I1104 00:15:33.041036 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_9/attention/output/dense/bias:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:15:33.041082 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_9/attention/output/LayerNorm/beta:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:15:33.041128 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_9/attention/output/LayerNorm/gamma:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:15:33.041173 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_9/intermediate/dense/kernel:0, shape = (768, 3072), *INIT_FROM_CKPT*
I1104 00:15:33.041220 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_9/intermediate/dense/bias:0, shape = (3072,), *INIT_FROM_CKPT*
I1104 00:15:33.041266 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_9/output/dense/kernel:0, shape = (3072, 768), *INIT_FROM_CKPT*
I1104 00:15:33.041313 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_9/output/dense/bias:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:15:33.041359 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_9/output/LayerNorm/beta:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:15:33.041405 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_9/output/LayerNorm/gamma:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:15:33.041450 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_10/attention/self/query/kernel:0, shape = (768, 768), *INIT_FROM_CKPT*
I1104 00:15:33.041498 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_10/attention/self/query/bias:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:15:33.041545 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_10/attention/self/key/kernel:0, shape = (768, 768), *INIT_FROM_CKPT*
I1104 00:15:33.041593 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_10/attention/self/key/bias:0, shape = (768,), *INIT_FROM_CKPT*
```

```
I1104 00:15:33.041639 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_10/attention/self/value/kernel:0, shape = (768, 768), *INIT_FROM_CKPT*
I1104 00:15:33.041687 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_10/attention/self/value/bias:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:15:33.041733 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_10/attention/output/dense/kernel:0, shape = (768, 768), *INIT_FROM_CKPT*
I1104 00:15:33.041781 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_10/attention/output/dense/bias:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:15:33.041828 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_10/attention/output/LayerNorm/beta:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:15:33.041874 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_10/attention/output/LayerNorm/gamma:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:15:33.041920 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_10/intermediate/dense/kernel:0, shape = (768, 3072), *INIT_FROM_CKPT*
I1104 00:15:33.041968 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_10/intermediate/dense/bias:0, shape = (3072,), *INIT_FROM_CKPT*
I1104 00:15:33.042015 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_10/output/dense/kernel:0, shape = (3072, 768), *INIT_FROM_CKPT*
I1104 00:15:33.042064 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_10/output/dense/bias:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:15:33.042110 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_10/output/LayerNorm/beta:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:15:33.042157 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_10/output/LayerNorm/gamma:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:15:33.042203 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_11/attention/self/query/kernel:0, shape = (768, 768), *INIT_FROM_CKPT*
I1104 00:15:33.042263 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_11/attention/self/query/bias:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:15:33.042310 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_11/attention/self/key/kernel:0, shape = (768, 768), *INIT_FROM_CKPT*
I1104 00:15:33.042360 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_11/attention/self/key/bias:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:15:33.042406 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_11/attention/self/value/kernel:0, shape = (768, 768), *INIT_FROM_CKPT*
I1104 00:15:33.042455 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_11/attention/self/value/bias:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:15:33.042501 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_11/attention/output/dense/kernel:0, shape = (768, 768), *INIT_FROM_CKPT*
I1104 00:15:33.042550 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_11/attention/output/dense/bias:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:15:33.042596 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_11/attention/output/LayerNorm/beta:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:15:33.042643 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_11/attention/output/LayerNorm/gamma:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:15:33.042688 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_11/intermediate/dense/kernel:0, shape = (768, 3072), *INIT_FROM_CKPT*
I1104 00:15:33.042737 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_11/intermediate/dense/bias:0, shape = (3072,), *INIT_FROM_CKPT*
I1104 00:15:33.042784 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_11/output/dense/kernel:0, shape = (3072, 768), *INIT_FROM_CKPT*
I1104 00:15:33.042832 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_11/output/dense/bias:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:15:33.042878 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_11/output/LayerNorm/beta:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:15:33.042924 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_11/output/LayerNorm/gamma:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:15:33.042970 139815656863488 run_classifier.py:669]   name = bert/pooler/dense/kernel:0, shape = (768, 768), *INIT_FROM_CKPT*
I1104 00:15:33.043018 139815656863488 run_classifier.py:669]   name = bert/pooler/dense/bias:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:15:33.043077 139815656863488 run_classifier.py:669]   name = output_weights:0, shape = (2, 768)
I1104 00:15:33.043138 139815656863488 run_classifier.py:669]   name = output_bias:0, shape = (2,)
W1104 00:15:33.043216 139815656863488 deprecation_wrapper.py:119] From /home/psakhamo/MyFolder/BERT_Exploration/bert/optimization.py:27: The name
tf.train.get_or_create_global_step is deprecated. Please use tf.compat.v1.train.get_or_create_global_step instead.

W1104 00:15:33.043682 139815656863488 deprecation_wrapper.py:119] From /home/psakhamo/MyFolder/BERT_Exploration/bert/optimization.py:32: The name
tf.train.polynomial_decay is deprecated. Please use tf.compat.v1.train.polynomial_decay instead.

W1104 00:15:33.046955 139815656863488 deprecation.py:323] From /home/psakhamo/.local/lib/python3.5/site-
packages/tensorflow/python/keras/optimizer_v2/learning_rate_schedule.py:409: div (from tensorflow.python.ops.math_ops) is deprecated and will be removed in a future
version.
Instructions for updating:
Deprecated in favor of operator or tf.math.divide.
W1104 00:15:33.165215 139815656863488 deprecation.py:323] From /home/psakhamo/.local/lib/python3.5/site-packages/tensorflow/python/ops/math_grad.py:1205:
add_dispatch_support.<locals>.wrapper (from tensorflow.python.ops.array_ops) is deprecated and will be removed in a future version.
Instructions for updating:
Use tf.where in 2.0, which has the same broadcast rule as np.where
I1104 00:15:38.043753 139815656863488 estimator.py:1147] Done calling model_fn.
I1104 00:15:38.044701 139815656863488 basic_session_run_hooks.py:541] Create CheckpointSaverHook.
I1104 00:15:40.029056 139815656863488 monitored_session.py:240] Graph was finalized.
2020-11-04 00:15:40.029324: I tensorflow/core/platform/cpu_feature_guard.cc:142] Your CPU supports instructions that this TensorFlow binary was not compiled to use: AVX2
FMA
2020-11-04 00:15:40.034059: I tensorflow/stream_executor/platform/default/dso_loader.cc:42] Successfully opened dynamic library libcuda.so.1
2020-11-04 00:15:40.102261: I tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:1005] successful NUMA node read from SysFS had negative value (-1), but there must
be at least one NUMA node, so returning NUMA node zero
2020-11-04 00:15:40.102809: I tensorflow/compiler/xla/service/service.cc:168] XLA service 0xcc95f10 executing computations on platform CUDA. Devices:
2020-11-04 00:15:40.102826: I tensorflow/compiler/xla/service/service.cc:175]   StreamExecutor device (0): GeForce GTX
1060 6GB, Compute Capability 6.1
2020-11-04 00:15:40.121054: I tensorflow/core/platform/profile_utils/cpu_utils.cc:94] CPU Frequency: 3600000000 Hz
2020-11-04 00:15:40.121960: I tensorflow/compiler/xla/service/service.cc:168] XLA service 0xc754480 executing computations on platform Host. Devices:
2020-11-04 00:15:40.122028: I tensorflow/compiler/xla/service/service.cc:175]   StreamExecutor device (0): <undefined>, <undefined>
2020-11-04 00:15:40.122348: I tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:1005] successful NUMA node read from SysFS had negative value (-1), but there must be at least one NUMA node, so
returning NUMA node zero
2020-11-04 00:15:40.123577: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1640] Found device 0 with properties:
name: GeForce GTX 1060 6GB major: 6 minor: 1 memoryClockRate(GHz): 1.7085
pciBusID: 0000:01:00.0
2020-11-04 00:15:40.124007: I tensorflow/stream_executor/platform/default/dso_loader.cc:42] Successfully opened dynamic library libcudart.so.10.0
2020-11-04 00:15:40.126252: I tensorflow/stream_executor/platform/default/dso_loader.cc:42] Successfully opened dynamic library libcublas.so.10.0
2020-11-04 00:15:40.128164: I tensorflow/stream_executor/platform/default/dso_loader.cc:42] Successfully opened dynamic library libcufft.so.10.0
2020-11-04 00:15:40.128741: I tensorflow/stream_executor/platform/default/dso_loader.cc:42] Successfully opened dynamic library libcurand.so.10.0
2020-11-04 00:15:40.131588: I tensorflow/stream_executor/platform/default/dso_loader.cc:42] Successfully opened dynamic library libcusolver.so.10.0
2020-11-04 00:15:40.133616: I tensorflow/stream_executor/platform/default/dso_loader.cc:42] Successfully opened dynamic library libcusparse.so.10.0
2020-11-04 00:15:40.139412: I tensorflow/stream_executor/platform/default/dso_loader.cc:42] Successfully opened dynamic library libcudnn.so.7
2020-11-04 00:15:40.139622: I tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:1005] successful NUMA node read from SysFS had negative value (-1), but there must be at least one NUMA node, so
returning NUMA node zero
2020-11-04 00:15:40.140828: I tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:1005] successful NUMA node read from SysFS had negative value (-1), but there must be at least one NUMA node, so
returning NUMA node zero
2020-11-04 00:15:40.141728: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1763] Adding visible gpu devices: 0
2020-11-04 00:15:40.141792: I tensorflow/stream_executor/platform/default/dso_loader.cc:42] Successfully opened dynamic library libcudart.so.10.0
2020-11-04 00:15:40.143310: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1181] Device interconnect StreamExecutor with strength 1 edge matrix:
2020-11-04 00:15:40.143338: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1187]      0
2020-11-04 00:15:40.143350: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1200] 0:   N
2020-11-04 00:15:40.143522: I tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:1005] successful NUMA node read from SysFS had negative value (-1), but there must be at least one NUMA node, so
returning NUMA node zero
2020-11-04 00:15:40.144496: I tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:1005] successful NUMA node read from SysFS had negative value (-1), but there must be at least one NUMA node, so
returning NUMA node zero
2020-11-04 00:15:40.145429: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1326] Created TensorFlow device (/job:localhost/replica:0/task:0/device:GPU:0 with 5175 MB memory) -> physical GPU
(device: 0, name: GeForce GTX 1060 6GB, pci bus id: 0000:01:00.0, compute capability: 6.1)
W1104 00:15:40.147133 139815656863488 deprecation.py:323] From /home/psakhamo/.local/lib/python3.5/site-packages/tensorflow/python/training/saver.py:1276: checkpoint_exists (from
tensorflow.python.training.checkpoint_management) is deprecated and will be removed in a future version.
Instructions for updating:
Use standard file APIs to check for files with this prefix.
I1104 00:15:40.148545 139815656863488 saver.py:1280] Restoring parameters from ./out/model.ckpt-0
W1104 00:15:42.169252 139815656863488 deprecation.py:323] From /home/psakhamo/.local/lib/python3.5/site-packages/tensorflow/python/training/saver.py:1066: get_checkpoint_mtimes (from
tensorflow.python.training.checkpoint_management) is deprecated and will be removed in a future version.
Instructions for updating:
Use standard file utilities to get mtimes.
```

```
2020-11-04 00:15:42.656733: W tensorflow/compiler/jit/mark_for_compilation_pass.cc:1412] (One-time warning): Not using XLA:CPU for cluster because envvar TF_XLA_FLAGS=--tf_xla_cpu_global_jit was not
set.  If you want XLA:CPU, either set that envvar, or use experimental_jit_scope to enable XLA:CPU.  To confirm that XLA is active, pass --vmodule=xla_compilation_cache=1 (as a proper command-line flag, not
via TF_XLA_FLAGS) or set the envvar XLA_FLAGS=--xla_hlo_profile.
I1104 00:15:42.714527 139815656863488 session_manager.py:500] Running local_init_op.
I1104 00:15:42.828996 139815656863488 session_manager.py:502] Done running local_init_op.
I1104 00:15:47.332123 139815656863488 basic_session_run_hooks.py:606] Saving checkpoints for 0 into ./out/model.ckpt.
2020-11-04 00:15:55.552387: I tensorflow/stream_executor/platform/default/dso_loader.cc:42] Successfully opened dynamic library libcublas.so.10.0
I1104 00:15:59.852509 139815656863488 tpu_estimator.py:2159] global_step/sec: 0.263558
I1104 00:15:59.852812 139815656863488 tpu_estimator.py:2160] examples/sec: 3.1627
I1104 00:16:00.169452 139815656863488 tpu_estimator.py:2159] global_step/sec: 3.15512
I1104 00:16:00.169637 139815656863488 tpu_estimator.py:2160] examples/sec: 37.8614
I1104 00:16:00.485787 139815656863488 tpu_estimator.py:2159] global_step/sec: 3.16163
I1104 00:16:00.485921 139815656863488 tpu_estimator.py:2160] examples/sec: 37.9395
I1104 00:16:00.801466 139815656863488 tpu_estimator.py:2159] global_step/sec: 3.16739
I1104 00:16:00.801585 139815656863488 tpu_estimator.py:2160] examples/sec: 38.0087
I1104 00:16:01.117604 139815656863488 tpu_estimator.py:2159] global_step/sec: 3.1633
I1104 00:16:01.117722 139815656863488 tpu_estimator.py:2160] examples/sec: 37.9596
I1104 00:16:01.433400 139815656863488 tpu_estimator.py:2159] global_step/sec: 3.16674
I1104 00:16:01.433646 139815656863488 tpu_estimator.py:2160] examples/sec: 38.0009
I1104 00:16:01.749959 139815656863488 tpu_estimator.py:2159] global_step/sec: 3.15874
I1104 00:16:01.750097 139815656863488 tpu_estimator.py:2160] examples/sec: 37.9049
I1104 00:16:02.065502 139815656863488 tpu_estimator.py:2159] global_step/sec: 3.16925
I1104 00:16:02.065647 139815656863488 tpu_estimator.py:2160] examples/sec: 38.031
I1104 00:16:02.382017 139815656863488 tpu_estimator.py:2159] global_step/sec: 3.15955
I1104 00:16:02.382158 139815656863488 tpu_estimator.py:2160] examples/sec: 37.9146
I1104 00:16:02.697543 139815656863488 tpu_estimator.py:2159] global_step/sec: 3.16907
I1104 00:16:
………
………..
I1104 00:20:51.791641 139815656863488 tpu_estimator.py:2159] global_step/sec: 3.1474
I1104 00:20:51.791836 139815656863488 tpu_estimator.py:2160] examples/sec: 37.7688
I1104 00:20:52.109416 139815656863488 tpu_estimator.py:2159] global_step/sec: 3.14707
I1104 00:20:52.109560 139815656863488 tpu_estimator.py:2160] examples/sec: 37.7648
I1104 00:20:52.427321 139815656863488 tpu_estimator.py:2159] global_step/sec: 3.14558
I1104 00:20:52.427457 139815656863488 tpu_estimator.py:2160] examples/sec: 37.7469
I1104 00:20:52.745221 139815656863488 tpu_estimator.py:2159] global_step/sec: 3.14536
I1104 00:20:52.745363 139815656863488 tpu_estimator.py:2160] examples/sec: 37.7443
I1104 00:20:52.745774 139815656863488 basic_session_run_hooks.py:606] Saving checkpoints for 917 into ./out/model.ckpt.
I1104 00:20:54.374748 139815656863488 estimator.py:368] Loss for final step: 0.25693655.
I1104 00:20:54.375319 139815656863488 error_handling.py:96] training_loop marked as finished
I1104 00:20:54.382089 139815656863488 run_classifier.py:487] Writing example 0 of 408
I1104 00:20:54.382594 139815656863488 run_classifier.py:461] *** Example ***
I1104 00:20:54.382662 139815656863488 run_classifier.py:462] guid: dev-1
I1104 00:20:54.382717 139815656863488 run_classifier.py:464] tokens: [CLS] he said the foods ##er ##vic ##e pie business doesn ' t fit the company ' s long - term growth strategy . [SEP] " the foods ##er ##vic
##e pie business does not fit our long - term growth strategy . [SEP]
I1104 00:20:54.382771 139815656863488 run_classifier.py:465] input_ids: 101 2002 2056 1996 9440 2121 7903 2063 11345 2449 2987 1005 1056 4906 1996 2194 1005 1055 2146 1011 2744 3930 5656 1012
102 1000 1996 9440 2121 7903 2063 11345 2449 2515 2025 4906 2256 2146 1011 2744 3930 5656 1012 102 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
I1104 00:20:54.382823 139815656863488 run_classifier.py:466] input_mask: 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
I1104 00:20:54.382873 139815656863488 run_classifier.py:467] segment_ids: 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
I1104 00:20:54.382912 139815656863488 run_classifier.py:468] label: 1 (id = 1)
I1104 00:20:54.383689 139815656863488 run_classifier.py:461] *** Example ***
I1104 00:20:54.383745 139815656863488 run_classifier.py:462] guid: dev-2
I1104 00:20:54.383801 139815656863488 run_classifier.py:464] tokens: [CLS] magna ##relli said ra ##cic ##ot hated the iraqi regime and looked forward to using his long years of training in the war . [SEP] his
wife said he was " 100 percent behind george bush " and looked forward to using his years of training in the war . [SEP]
I1104 00:20:54.383855 139815656863488 run_classifier.py:465] input_ids: 101 20201 22948 2056 10958 19053 4140 6283 1996 8956 6939 1998 2246 2830 2000 2478 2010 2146 2086 1997 2731 1999 1996
2162 1012 102 2010 2564 2056 2002 2001 1000 2531 3867 2369 2577 5747 1000 1998 2246 2830 2000 2478 2010 2086 1997 2731 1999 1996 2162 1012 102 0 0 0 0 0 0 0 0 0 0 0 0
I1104 00:20:54.383905 139815656863488 run_classifier.py:466] input_mask: 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0
I1104 00:20:54.383954 139815656863488 run_classifier.py:467] segment_ids: 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0
I1104 00:20:54.383993 139815656863488 run_classifier.py:468] label: 0 (id = 0)
I1104 00:20:54.384781 139815656863488 run_classifier.py:461] *** Example ***
I1104 00:20:54.384836 139815656863488 run_classifier.py:462] guid: dev-3
I1104 00:20:54.384895 139815656863488 run_classifier.py:464] tokens: [CLS] the dollar was at 116 . 92 yen against the yen , flat on the session , and at 1 . 289 ##1 against the swiss fran ##c , also flat [SEP]
the dollar was at 116 . 78 yen jp ##y = , virtually flat on the session , and at 1 . 287 ##1 against the swiss fran ##c ch [SEP]
I1104 00:20:54.384949 139815656863488 run_classifier.py:465] input_ids: 101 1996 7922 2001 2012 12904 1012 6227 18371 2114 1996 18371 1010 4257 2006 1996 5219 1010 1998 2012 1015 1012 27054
2487 2114 1996 5364 23151 2278 1010 2036 4257 102 1996 7922 2001 2012 12904 1012 6275 18371 16545 2100 1027 1010 8990 4257 2006 1996 5219 1010 1998 2012 1015 1012 23090 2487 2114 1996
5364 23151 2278 10381 102
I1104 00:20:54.385000 139815656863488 run_classifier.py:466] input_mask: 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
I1104 00:20:54.385049 139815656863488 run_classifier.py:467] segment_ids: 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
I1104 00:20:54.385088 139815656863488 run_classifier.py:468] label: 0 (id = 0)
I1104 00:20:54.385728 139815656863488 run_classifier.py:461] *** Example ***
I1104 00:20:54.385783 139815656863488 run_classifier.py:462] guid: dev-4
I1104 00:20:54.385837 139815656863488 run_classifier.py:464] tokens: [CLS] the afl - ci ##o is waiting until october to decide if it will end ##ors ##e a candidate . [SEP] the afl - ci ##o announced wednesday
that it will decide in october whether to end ##ors ##e a candidate before the primaries . [SEP]
I1104 00:20:54.385889 139815656863488 run_classifier.py:465] input_ids: 101 1996 10028 1011 25022 2080 2003 3403 2127 2255 2000 5630 2065 2009 2097 2203 5668 2063 1037 4018 1012 102 1996
10028 1011 25022 2080 2623 9317 2008 2009 2097 5630 1999 2255 3251 2000 2203 5668 2063 1037 4018 2077 1996 27419 1012 102 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
I1104 00:20:54.385940 139815656863488 run_classifier.py:466] input_mask: 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
I1104 00:20:54.385989 139815656863488 run_classifier.py:467] segment_ids: 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
I1104 00:20:54.386029 139815656863488 run_classifier.py:468] label: 1 (id = 1)
I1104 00:20:54.386603 139815656863488 run_classifier.py:461] *** Example ***
I1104 00:20:54.386658 139815656863488 run_classifier.py:462] guid: dev-5
I1104 00:20:54.386725 139815656863488 run_classifier.py:464] tokens: [CLS] no dates have been set for the civil or the criminal trial . [SEP] no dates have been set for the criminal or civil cases , but shan ##ley
has pleaded not guilty . [SEP]
I1104 00:20:54.386806 139815656863488 run_classifier.py:465] input_ids: 101 2053 5246 2031 2042 2275 2005 1996 2942 2030 1996 4735 3979 1012 102 2053 5246 2031 2042 2275 2005 1996 4735 2030
2942 3572 1010 2021 17137 3051 2038 12254 2025 5905 1012 102 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
I1104 00:20:54.386884 139815656863488 run_classifier.py:466] input_mask: 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
I1104 00:20:54.386960 139815656863488 run_classifier.py:467] segment_ids: 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
I1104 00:20:54.387026 139815656863488 run_classifier.py:468] label: 0 (id = 0)
I1104 00:20:54.653697 139815656863488 run_classifier.py:898] ***** Running evaluation *****
I1104 00:20:54.653815 139815656863488 run_classifier.py:901]   Num examples = 408 (408 actual, 0 padding)
I1104 00:20:54.653875 139815656863488 run_classifier.py:902]   Batch size = 8

I1104 00:20:54.679294 139815656863488 estimator.py:1145] Calling model_fn.

I1104 00:20:54.679490 139815656863488 tpu_estimator.py:2965] Running eval on CPU
I1104 00:20:54.679872 139815656863488 run_classifier.py:627] *** Features ***
I1104 00:20:54.680013 139815656863488 run_classifier.py:629]   name = input_ids, shape = (?, 64)
I1104 00:20:54.680112 139815656863488 run_classifier.py:629]   name = input_mask, shape = (?, 64)
I1104 00:20:54.680172 139815656863488 run_classifier.py:629]   name = is_real_example, shape = (?,)
I1104 00:20:54.680227 139815656863488 run_classifier.py:629]   name = label_ids, shape = (?,)
I1104 00:20:54.680283 139815656863488 run_classifier.py:629]   name = segment_ids, shape = (?, 64)
I1104 00:20:56.674241 139815656863488 run_classifier.py:663] **** Trainable Variables ****
I1104 00:20:56.674396 139815656863488 run_classifier.py:669]   name = bert/embeddings/word_embeddings:0, shape = (30522, 768), *INIT_FROM_CKPT*
I1104 00:20:56.674500 139815656863488 run_classifier.py:669]   name = bert/embeddings/token_type_embeddings:0, shape = (2, 768), *INIT_FROM_CKPT*
I1104 00:20:56.674585 139815656863488 run_classifier.py:669]   name = bert/embeddings/position_embeddings:0, shape = (512, 768), *INIT_FROM_CKPT*
I1104 00:20:56.674666 139815656863488 run_classifier.py:669]   name = bert/embeddings/LayerNorm/beta:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:20:56.674728 139815656863488 run_classifier.py:669]   name = bert/embeddings/LayerNorm/gamma:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:20:56.674790 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_0/attention/self/query/kernel:0, shape = (768, 768), *INIT_FROM_CKPT*
I1104 00:20:56.674867 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_0/attention/self/query/bias:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:20:56.674914 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_0/attention/self/key/kernel:0, shape = (768, 768), *INIT_FROM_CKPT*
I1104 00:20:56.674962 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_0/attention/self/key/bias:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:20:56.675009 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_0/attention/self/value/kernel:0, shape = (768, 768), *INIT_FROM_CKPT*
I1104 00:20:56.675058 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_0/attention/self/value/bias:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:20:56.675105 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_0/attention/output/dense/kernel:0, shape = (768, 768), *INIT_FROM_CKPT*
```

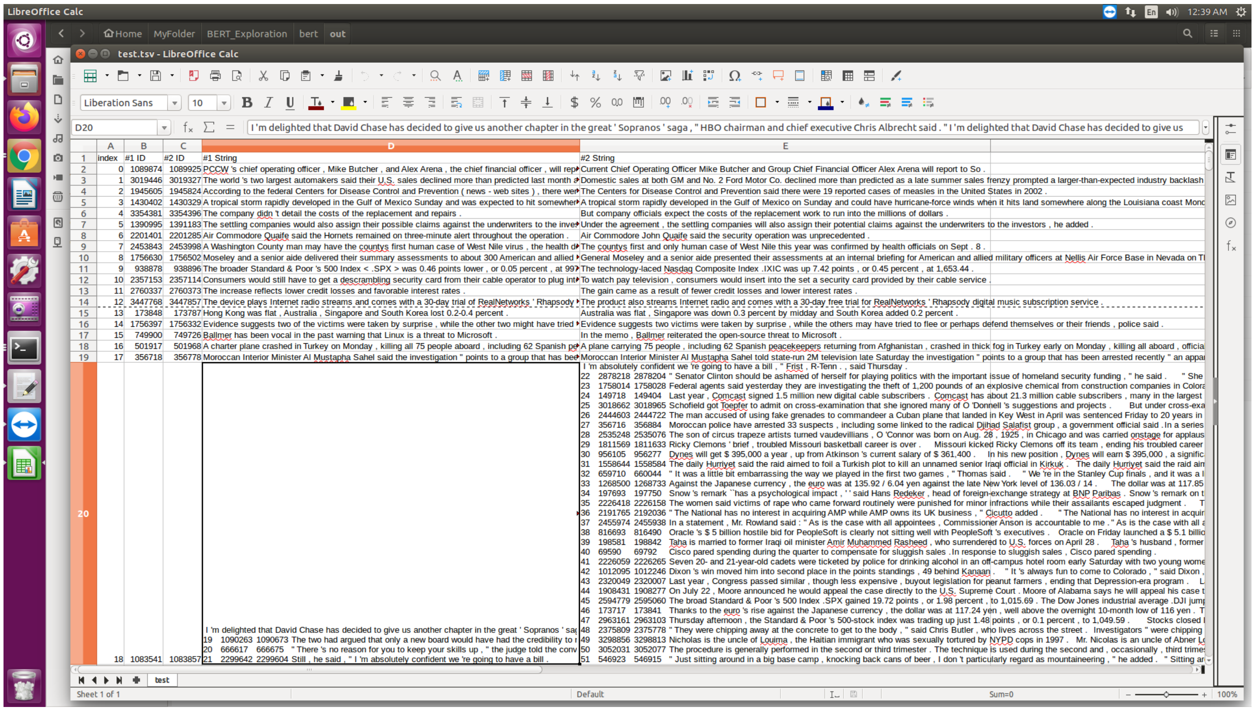
```
I1104 00:20:56.675154 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_0/attention/output/dense/bias:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:20:56.675200 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_0/attention/output/LayerNorm/beta:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:20:56.675246 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_0/attention/output/LayerNorm/gamma:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:20:56.675292 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_0/intermediate/dense/kernel:0, shape = (768, 3072), *INIT_FROM_CKPT*
I1104 00:20:56.675341 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_0/intermediate/dense/bias:0, shape = (3072,), *INIT_FROM_CKPT*
I1104 00:20:56.678489 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_4/output/dense/bias:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:20:56.678535 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_4/output/LayerNorm/beta:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:20:56.678580 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_4/output/LayerNorm/gamma:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:20:56.678626 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_5/attention/self/query/kernel:0, shape = (768, 768), *INIT_FROM_CKPT*
I1104 00:20:56.678674 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_5/attention/self/query/bias:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:20:56.678720 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_5/attention/self/key/kernel:0, shape = (768, 768), *INIT_FROM_CKPT*
I1104 00:20:56.678769 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_5/attention/self/key/bias:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:20:56.678815 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_5/attention/self/value/kernel:0, shape = (768, 768), *INIT_FROM_CKPT*
I1104 00:20:56.678863 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_5/attention/self/value/bias:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:20:56.678910 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_5/attention/output/dense/kernel:0, shape = (768, 768), *INIT_FROM_CKPT*
I1104 00:20:56.678957 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_5/attention/output/dense/bias:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:20:56.679003 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_5/attention/output/LayerNorm/beta:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:20:56.679049 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_5/attention/output/LayerNorm/gamma:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:20:56.679094 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_5/intermediate/dense/kernel:0, shape = (768, 3072), *INIT_FROM_CKPT*
I1104 00:20:56.679142 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_5/intermediate/dense/bias:0, shape = (3072,), *INIT_FROM_CKPT*
I1104 00:20:56.679188 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_5/output/dense/kernel:0, shape = (3072, 768), *INIT_FROM_CKPT*
I1104 00:20:56.679236 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_5/output/dense/bias:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:20:56.679282 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_5/output/LayerNorm/beta:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:20:56.679327 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_5/output/LayerNorm/gamma:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:20:56.679373 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_6/attention/self/query/kernel:0, shape = (768, 768), *INIT_FROM_CKPT*
I1104 00:20:56.679421 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_6/attention/self/query/bias:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:20:56.679467 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_6/attention/self/key/kernel:0, shape = (768, 768), *INIT_FROM_CKPT*
I1104 00:20:56.679514 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_6/attention/self/key/bias:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:20:56.679560 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_6/attention/self/value/kernel:0, shape = (768, 768), *INIT_FROM_CKPT*
I1104 00:20:56.679607 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_6/attention/self/value/bias:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:20:56.679653 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_6/attention/output/dense/kernel:0, shape = (768, 768), *INIT_FROM_CKPT*
I1104 00:20:56.679701 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_6/attention/output/dense/bias:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:20:56.679747 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_6/attention/output/LayerNorm/beta:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:20:56.679793 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_6/attention/output/LayerNorm/gamma:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:20:56.679838 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_6/intermediate/dense/kernel:0, shape = (768, 3072), *INIT_FROM_CKPT*
I1104 00:20:56.679886 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_6/intermediate/dense/bias:0, shape = (3072,), *INIT_FROM_CKPT*
I1104 00:20:56.679933 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_6/output/dense/kernel:0, shape = (3072, 768), *INIT_FROM_CKPT*
I1104 00:20:56.679980 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_6/output/dense/bias:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:20:56.680026 139815656863488 run_classifier.py:669]   name = bert/encoder/layer_6/output/LayerNorm/beta:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:20:56.680071 1…
…….
……..
I1104 00:20:56.683881 139815656863488 run_classifier.py:669]   name = bert/pooler/dense/bias:0, shape = (768,), *INIT_FROM_CKPT*
I1104 00:20:56.683939 139815656863488 run_classifier.py:669]   name = output_weights:0, shape = (2, 768)
I1104 00:20:56.683999 139815656863488 run_classifier.py:669]   name = output_bias:0, shape = (2,)
W1104 00:20:56.685064 139815656863488 deprecation_wrapper.py:119] From run_classifier.py:686: The name tf.metrics.accuracy is deprecated. Please use tf.compat.v1.metrics.accuracy instead.

W1104 00:20:56.724731 139815656863488 deprecation_wrapper.py:119] From run_classifier.py:688: The name tf.metrics.mean is deprecated. Please use tf.compat.v1.metrics.mean instead.

I1104 00:20:56.761431 139815656863488 estimator.py:1147] Done calling model_fn.
I1104 00:20:56.774543 139815656863488 evaluation.py:255] Starting evaluation at 2020-11-04T00:20:56Z
I1104 00:20:57.148565 139815656863488 monitored_session.py:240] Graph was finalized.
2020-11-04 00:20:57.148995: I tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:1005] successful NUMA node read from SysFS had negative value (-1), but there must be at least one NUMA node, so returning NUMA node
zero
2020-11-04 00:20:57.149430: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1640] Found device 0 with properties:
name: GeForce GTX 1060 6GB major: 6 minor: 1 memoryClockRate(GHz): 1.7085
pciBusID: 0000:01:00.0
2020-11-04 00:20:57.149498: I tensorflow/stream_executor/platform/default/dso_loader.cc:42] Successfully opened dynamic library libcudart.so.10.0
2020-11-04 00:20:57.149527: I tensorflow/stream_executor/platform/default/dso_loader.cc:42] Successfully opened dynamic library libcublas.so.10.0
2020-11-04 00:20:57.149536: I tensorflow/stream_executor/platform/default/dso_loader.cc:42] Successfully opened dynamic library libcufft.so.10.0
2020-11-04 00:20:57.149544: I tensorflow/stream_executor/platform/default/dso_loader.cc:42] Successfully opened dynamic library libcurand.so.10.0
2020-11-04 00:20:57.149552: I tensorflow/stream_executor/platform/default/dso_loader.cc:42] Successfully opened dynamic library libcusolver.so.10.0
2020-11-04 00:20:57.149578: I tensorflow/stream_executor/platform/default/dso_loader.cc:42] Successfully opened dynamic library libcusparse.so.10.0
2020-11-04 00:20:57.149603: I tensorflow/stream_executor/platform/default/dso_loader.cc:42] Successfully opened dynamic library libcudnn.so.7
2020-11-04 00:20:57.149663: I tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:1005] successful NUMA node read from SysFS had negative value (-1), but there must be at least one NUMA node, so returning NUMA node
zero
2020-11-04 00:20:57.150077: I tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:1005] successful NUMA node read from SysFS had negative value (-1), but there must be at least one NUMA node, so returning NUMA node
zero
2020-11-04 00:20:57.150462: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1763] Adding visible gpu devices: 0
2020-11-04 00:20:57.150499: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1181] Device interconnect StreamExecutor with strength 1 edge matrix:
2020-11-04 00:20:57.150521: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1187]      0
2020-11-04 00:20:57.150542: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1200] 0:   N
2020-11-04 00:20:57.150630: I tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:1005] successful NUMA node read from SysFS had negative value (-1), but there must be at least one NUMA node, so returning NUMA node
zero
2020-11-04 00:20:57.151036: I tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:1005] successful NUMA node read from SysFS had negative value (-1), but there must be at least one NUMA node, so returning NUMA node
zero
2020-11-04 00:20:57.151413: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1326] Created TensorFlow device (/job:localhost/replica:0/task:0/device:GPU:0 with 5175 MB memory) -> physical GPU (device: 0, name:
GeForce GTX 1060 6GB, pci bus id: 0000:01:00.0, compute capability: 6.1)
I1104 00:20:57.152237 139815656863488 saver.py:1280] Restoring parameters from ./out/model.ckpt-917
I1104 00:20:57.714351 139815656863488 session_manager.py:500] Running local_init_op.
I1104 00:20:57.757349 139815656863488 session_manager.py:502] Done running local_init_op.
I1104 00:21:01.072486 139815656863488 evaluation.py:275] Finished evaluation at 2020-11-04-00:21:01
I1104 00:21:01.072738 139815656863488 estimator.py:2039] Saving dict for global step 917: eval_accuracy = 0.8504902, eval_loss = 0.5313762, global_step = 917, loss = 0.5313762
I1104 00:21:01.435395 139815656863488 estimator.py:2099] Saving 'checkpoint_path' summary for global step 917: ./out/model.ckpt-917
I1104 00:21:01.435890 139815656863488 error_handling.py:96] evaluation_loop marked as finished
I1104 00:21:01.436035 139815656863488 run_classifier.py:923] ***** Eval results *****
I1104 00:21:01.436127 139815656863488 run_classifier.py:925]   eval_accuracy = 0.8504902
I1104 00:21:01.436449 139815656863488 run_classifier.py:925]   eval_loss = 0.5313762
I1104 00:21:01.436551 139815656863488 run_classifier.py:925]   global_step = 917
I1104 00:21:01.436655 139815656863488 run_classifier.py:925]   loss = 0.5313762
```

Prediction on Trained Classifier

Test input: test.tsv (sample)



D20 = I 'm delighted that David Chase has decided to give us another chapter in the great ' Sopranos ' saga , " HBO chairman and chief executive Chris Albrecht said . " I 'm delighted that David Chase has decided to give us

	A	B	C	D	E
1	index	#1 ID	#2 ID	#1 String	#2 String
2	0	1089874	1089925	PCCW 's chief operating officer , Mike Butcher , and Alex Arena , the chief financial officer , will rep	Current Chief Operating Officer Mike Butcher and Group Chief Financial Officer Alex Arena will report to So .
3	1	3019446	3019327	The world 's two largest automakers said their U.S. sales declined more than predicted last month a	Domestic sales at both GM and No. 2 Ford Motor Co. declined more than predicted as a late summer sales frenzy prompted a larger-than-expected industry backlash
4	2	1945605	1945824	According to the federal Centers for Disease Control and Prevention (news - web sites) , there wer	The Centers for Disease Control and Prevention said there were 19 reported cases of measles in the United States in 2002 .
5	3	1430402	1430329	A tropical storm rapidly developed in the Gulf of Mexico Sunday and was expected to hit somewher	A tropical storm rapidly developed in the Gulf of Mexico on Sunday and could have hurricane-force winds when it hits land somewhere along the Louisiana coast Monc
6	4	3354381	3354396	The company didn 't detail the costs of the replacement and repairs .	But company officials expect the costs of the replacement work to run into the millions of dollars .
7	5	1390995	1391183	The settling companies would also assign their possible claims against the underwriters to the inve	Under the agreement , the settling companies will also assign their potential claims against the underwriters to the investors , he added .
8	6	2201401	2201285	Air Commodore Quaife said the Hornets remained on three-minute alert throughout the operation .	Air Commodore John Quaife said the security operation was unprecedented .
9	7	2453843	2453998	A Washington County man may have the countys first human case of West Nile virus , the health d	The countys first and only human case of West Nile this year was confirmed by health officials on Sept . 8 .
10	8	1756630	1756502	Moseley and a senior aide delivered their summary assessments to about 300 American and allied	General Moseley and a senior aide presented their assessments at an internal briefing for American and allied military officers at Nellis Air Force Base in Nevada on Tl
11	9	938878	938896	The broader Standard & Poor 's 500 Index < .SPX > was 0.46 points lower , or 0.05 percent , at 997	The technology-laced Nasdaq Composite Index .IXIC was up 7.42 points , or 0.45 percent , at 1,653.44 .
12	10	2357153	2357114	Consumers would still have to get a descrambling security card from their cable operator to plug int	To watch pay television , consumers would insert into the set a security card provided by their cable service .
13	11	2760337	2760373	The increase reflects lower credit losses and favorable interest rates .	The gain came as a result of fewer credit losses and lower interest rates .
14	12	3447768	3447857	The device plays Internet radio streams and comes with a 30-day trial of RealNetworks ' Rhapsody	The product also streams Internet radio and comes with a 30-day free trial for RealNetworks ' Rhapsody digital music subscription service .
15	13	173848	173787	Hong Kong was flat , Australia , Singapore and South Korea lost 0.2-0.4 percent .	Australia was flat , Singapore was down 0.3 percent by midday and South Korea added 0.2 percent .
16	14	1756397	1756332	Evidence suggests two of the victims were taken by surprise , while the other two might have tried	Evidence suggests two victims were taken by surprise , while the others may have tried to flee or perhaps defend themselves or their friends , police said .
17	15	749900	749726	Ballmer has been vocal in the past warning that Linux is a threat to Microsoft .	In the memo , Ballmer reiterated the open-source threat to Microsoft .
18	16	501917	501968	A charter plane crashed in Turkey on Monday , killing all 75 people aboard , including 62 Spanish pe	A plane carrying 75 people , including 62 Spanish peacekeepers returning from Afghanistan , crashed in thick fog in Turkey early on Monday , killing all aboard , officia
19	17	356718	356778	Moroccan Interior Minister Al Mustapha Sahel said the investigation " points to a group that has bee	Moroccan Interior Minister Al Mustapha Sahel told state-run 2M television late Saturday the investigation " points to a group that has been arrested recently " an appai
20	18	1083541	1083857	I 'm delighted that David Chase has decided to give us another chapter in the great ' Sopranos ' sag 19 1090263 1090673 The two had argued that only a new board would have had the credibility to r 20 666617 666675 " There 's no reason for you to keep your skills up , " the judge told the conv 21 2299642 2299604 Still , he said , " I 'm absolutely confident we 're going to have a bill .	I 'm absolutely confident we 're going to have a bill , " Frist , R-Tenn . , said Thursday . 22 2878218 2878204 " Senator Clinton should be ashamed of herself for playing politics with the important issue of homeland security funding , " he said . " She 23 1758014 1758028 Federal agents said yesterday they are investigating the theft of 1,200 pounds of an explosive chemical from construction companies in Colora 24 149718 149404 Last year , Comcast signed 1.5 million new digital cable subscribers . Comcast has about 21.3 million cable subscribers , many in the largest 25 3018662 3018965 Schofield got Toepfer to admit on cross-examination that she ignored many of O 'Donnell 's suggestions and projects . But under cross-exa 26 2444603 2444722 The man accused of using fake grenades to commandeer a Cuban plane that landed in Key West in April was sentenced Friday to 20 years in 27 356716 356884 Moroccan police have arrested 33 suspects , including some linked to the radical Djihad Salafist group , a government official said .In a series 28 2535248 2535076 The son of circus trapeze artists turned vaudevillians , O 'Connor was born on Aug. 28 , 1925 , in Chicago and was carried onstage for applaus 29 1811569 1811633 Ricky Clemons ' brief , troubled Missouri basketball career is over . Missouri kicked Ricky Clemons off its team , ending his troubled career 30 956105 956277 Dynes will get $ 395,000 a year , up from Atkinson 's current salary of $ 361,400 . In his new position , Dynes will earn $ 395,000 , a signific 31 1558644 1558584 The daily Hurriyet said the raid aimed to foil a Turkish plot to kill an unnamed senior Iraqi official in Kirkuk . The daily Hurriyet said the raid aim 32 659710 660044 " It was a little bit embarrassing the way we played in the first two games , " Thomas said . " We 're in the Stanley Cup finals , and it was a l 33 1268500 1268733 Against the Japanese currency , the euro was at 135.92 / 6.04 yen against the late New York level of 136.03 / 14 . The dollar was at 117.85 34 197693 197750 Snow 's remark ``has a psychological impact , ' ' said Hans Redeker , head of foreign-exchange strategy at BNP Paribas . Snow 's remark on t 35 2226418 2226158 The women said victims of rape who came forward routinely were punished for minor infractions while their assailants escaped judgment . T 36 2191765 2192036 " The National has no interest in acquiring AMP while AMP owns its UK business , " Cicutto added . " The National has no interest in acquir 37 2455974 2455938 In a statement , Mr. Rowland said : " As is the case with all appointees , Commissioner Anson is accountable to me . " As is the case with all a 38 816693 816490 Oracle 's $ 5 billion hostile bid for PeopleSoft is clearly not sitting well with PeopleSoft 's executives . Oracle on Friday launched a $ 5.1 billio 39 198581 198842 Taha is married to former Iraqi oil minister Amir Muhammed Rasheed , who surrendered to U.S. forces on April 28 . Taha 's husband , former 40 69590 69792 Cisco pared spending during the quarter to compensate for sluggish sales .In response to sluggish sales , Cisco pared spending . 41 2226059 2226265 Seven 20- and 21-year-old cadets were ticketed by police for drinking alcohol in an off-campus hotel room early Saturday with two young wome 42 1012095 1012246 Dixon 's win moved him into second place in the points standings , 49 behind Kanaan . " It 's always fun to come to Colorado , " said Dixon , 43 2320049 2320007 Last year , Congress passed similar , though less expensive , buyout legislation for peanut farmers , ending that Depression-era program . L 44 1908431 1908277 On July 22 , Moore announced he would appeal the case directly to the U.S. Supreme Court . Moore of Alabama says he will appeal his case t 45 2594779 2595060 The broad Standard & Poor 's 500 Index .SPX gained 19.72 points , or 1.98 percent , to 1,015.69 . The Dow Jones industrial average .DJI jump 46 173717 173841 Thanks to the euro 's rise against the Japanese currency , the dollar was at 117.24 yen , well above the overnight 10-month low of 116 yen . T 47 2963161 2963103 Thursday afternoon , the Standard & Poor 's 500-stock index was trading up just 1.48 points , or 0.1 percent , to 1,049.59 . Stocks closed I 48 2375809 2375778 " They were chipping away at the concrete to get to the body , " said Chris Butler , who lives across the street . Investigators " were chipping 49 3298856 3298813 Nicholas is the uncle of Louima , the Haitian immigrant who was sexually tortured by NYPD cops in 1997 . Mr. Nicolas is an uncle of Abner Lo 50 3052031 3052077 The procedure is generally performed in the second or third trimester . The technique is used during the second and , occasionally , third trimes 51 546923 546915 " Just sitting around in a big base camp , knocking back cans of beer , I don 't particularly regard as mountaineering , " he added . " Sitting ar

```
python3 run_classifier.py --task_name=MRPC --do_predict=true --data_dir=$GLUE_DIR/MRPC --vocab_file=$BERT_BASE_DIR/vocab.txt --bert_config_file=$BERT_BASE_DIR/bert_config.json --init_checkpoint=$TRAINED_CLASSIFIER --max_seq_length=128 --output_dir=./out/
```

```
params are not passed to Estimator.
I1104 00:34:43.799808 140352953538304 estimator.py:209] Using config: {'_tf_random_seed': None, '_master': '', '_cluster_spec': <tensorflow.python.training.server_lib.ClusterSpec object at 0x7fa6175bde10>,
'_keep_checkpoint_every_n_hours': 10000, '_global_id_in_cluster': 0, '_save_summary_steps': 100, '_train_distribute': None, '_service': None, '_device_fn': None, '_keep_checkpoint_max': 5,
'_save_checkpoints_steps': 1000, '_task_id': 0, '_save_checkpoints_secs': None, '_tpu_config': TPUConfig(iterations_per_loop=1000, num_shards=8, num_cores_per_replica=None,
per_host_input_for_training=3, tpu_job_name=None, initial_infeed_sleep_secs=None, input_partition_dims=None, eval_training_input_configuration=2), '_evaluation_master': '', '_eval_distribute': None,
'_session_config': allow_soft_placement: true
graph_options {
  rewrite_options {
    meta_optimizer_iterations: ONE
  }
}
, '_log_step_count_steps': None, '_is_chief': True, '_model_dir': './out/', '_cluster': None, '_experimental_distribute': None, '_protocol': None, '_experimental_max_worker_delay_secs': None,
'_num_worker_replicas': 1, '_num_ps_replicas': 0, '_task_type': 'worker'}
I1104 00:34:43.800058 140352953538304 tpu_context.py:209] _TPUContext: eval_on_tpu True
W1104 00:34:43.800213 140352953538304 tpu_context.py:211] eval_on_tpu ignored because use_tpu is False.
W1104 00:34:43.800310 140352953538304 deprecation_wrapper.py:119] From run_classifier.py:199: The name tf.gfile.Open is deprecated. Please use tf.io.gfile.GFile instead.

W1104 00:34:43.812769 140352953538304 deprecation_wrapper.py:119] From run_classifier.py:483: The name tf.python_io.TFRecordWriter is deprecated. Please use tf.io.TFRecordWriter instead.

W1104 00:34:43.813036 140352953538304 deprecation_wrapper.py:119] From run_classifier.py:487: The name tf.logging.info is deprecated. Please use tf.compat.v1.logging.info instead.

I1104 00:34:43.813102 140352953538304 run_classifier.py:487] Writing example 0 of 1725
I1104 00:34:43.813711 140352953538304 run_classifier.py:461] *** Example ***
I1104 00:34:43.813779 140352953538304 run_classifier.py:462] guid: test-1
I1104 00:34:43.813863 140352953538304 run_classifier.py:464] tokens: [CLS] pc ##c ##w ' s chief operating officer , mike butcher , and alex arena , the chief financial officer , will report directly to mr so . [SEP]
current chief operating officer mike butcher and group chief financial officer alex arena will report to so . [SEP]
I1104 00:34:43.813951 140352953538304 run_classifier.py:465] input_ids: 101 7473 2278 2860 1005 1055 2708 4082 2961 1010 3505 14998 1010 1998 4074 5196 1010 1996 2708 3361 2961 1010 2097 3189
3495 2000 2720 2061 1012 102 2783 2708 4082 2961 3505 14998 1998 2177 2708 3361 2961 4074 5196 2097 3189 2000 2061 1012 102 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
I1104 00:34:43.814038 140352953538304 run_classifier.py:466] input_mask: 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
I1104 00:34:43.814096 140352953538304 run_classifier.py:467] segment_ids: 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
I1104 00:34:43.814136 140352953538304 run_classifier.py:468] label: 0 (id = 0)
I1104 00:34:43.815136 140352953538304 run_classifier.py:461] *** Example ***
I1104 00:34:43.815194 140352953538304 run_classifier.py:462] guid: test-2
I1104 00:34:43.815255 140352953538304 run_classifier.py:464] tokens: [CLS] the world ' s two largest auto ##makers said their u . s . sales declined more than predicted last month as a late summer sales
frenzy caused more of an industry backlash than expected . [SEP] domestic sales at both gm and no . 2 ford motor co . declined more than predicted as a late summer sales frenzy prompted a larger - than -
expected industry backlash . [SEP]
I1104 00:34:43.815318 140352953538304 run_classifier.py:465] input_ids: 101 1996 2088 1005 1055 2048 2922 8285 12088 2056 2037 1057 1012 1055 1012 4341 6430 2062 2084 10173 2197 3204 2004
1037 2397 2621 4341 21517 3303 2062 1997 2019 3068 25748 2084 3517 1012 102 4968 4341 2012 2119 13938 1998 2053 1012 1016 4811 5013 2522 1012 6430 2062 2084 10173 2004 1037 2397 2621
4341 21517 9469 1037 3469 1011 2084 1011 3517 3068 25748 1012 102 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

```
I1104 00:34:43.815379 140352953538304 run_classifier.py:466] input_mask: 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
I1104 00:34:43.815437 140352953538304 run_classifier.py:467] segment_ids: 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
I1104 00:34:43.815478 140352953538304 run_classifier.py:468] label: 0 (id = 0)
I1104 00:34:43.816311 140352953538304 run_classifier.py:461] *** Example ***
I1104 00:34:43.816367 140352953538304 run_classifier.py:462] guid: test-3
I1104 00:34:43.816425 140352953538304 run_classifier.py:464] tokens: [CLS] according to the federal centers for disease control and prevention ( news - web sites ) , there were 19 reported cases of me ##as
##les in the united states in 2002 . [SEP] the centers for disease control and prevention said there were 19 reported cases of me ##as ##les in the united states in 2002 . [SEP]
I1104 00:34:43.816487 140352953538304 run_classifier.py:465] input_ids: 101 2429 2000 1996 2976 6401 2005 4295 2491 1998 9740 1006 2739 1011 4773 4573 1007 1010 2045 2020 2539 2
```

```
Successfully opened dynamic library libcudart.so.10.0
2020-11-04 00:34:47.514045: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1181] Device interconnect StreamExecutor with strength 1 edge matrix:
2020-11-04 00:34:47.514058: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1187]      0
2020-11-04 00:34:47.514079: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1200] 0:   N
2020-11-04 00:34:47.514176: I tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:1005] successful NUMA node read from SysFS had negative value (-1), but there must be at least one NUMA node, so
returning NUMA node zero
2020-11-04 00:34:47.514660: I tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:1005] successful NUMA node read from SysFS had negative value (-1), but there must be at least one NUMA node, so
returning NUMA node zero
2020-11-04 00:34:47.515175: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1326] Created TensorFlow device (/job:localhost/replica:0/task:0/device:GPU:0 with 5141 MB memory) -> physical GPU
(device: 0, name: GeForce GTX 1060 6GB, pci bus id: 0000:01:00.0, compute capability: 6.1)
W1104 00:34:47.515968 140352953538304 deprecation.py:323] From /home/psakhamo/.local/lib/python3.5/site-packages/tensorflow/python/training/saver.py:1276: checkpoint_exists (from
tensorflow.python.training.checkpoint_management) is deprecated and will be removed in a future version.
Instructions for updating:
Use standard file APIs to check for files with this prefix.
I1104 00:34:47.516753 140352953538304 saver.py:1280] Restoring parameters from ./out/model.ckpt-917
2020-11-04 00:34:48.541594: W tensorflow/compiler/jit/mark_for_compilation_pass.cc:1412] (One-time warning): Not using XLA:CPU for cluster because envvar TF_XLA_FLAGS=--tf_xla_cpu_global_jit was not
set.  If you want XLA:CPU, either set that envvar, or use experimental_jit_scope to enable XLA:CPU.  To confirm that XLA is active, pass --vmodule=xla_compilation_cache=1 (as a proper command-line flag, not
via TF_XLA_FLAGS) or set the envvar XLA_FLAGS=--xla_hlo_profile.
I1104 00:34:48.560247 140352953538304 session_manager.py:500] Running local_init_op.
I1104 00:34:48.592525 140352953538304 session_manager.py:502] Done running local_init_op.
2020-11-04 00:34:49.048319: I tensorflow/stream_executor/platform/default/dso_loader.cc:42] Successfully opened dynamic library libcublas.so.10.0
I1104 00:35:08.415303 140352953538304 error_handling.py:96] prediction_loop marked as finished
I1104 00:35:08.415478 140352953538304 error_handling.py:96] prediction_loop marked as finished
```

Sample output: (test_results.tsv)

```
0.06499615   0.9350039

0.052415397 0.94758457

0.050007023 0.94999295

0.05509066    0.9449093

0.793359        0.20664103

0.047772396.  0.9522276
```