





JOIN NEWSLETTER

Datasets

Intent Recognition with BERT using Keras and TensorFlow 2

Resources

TL;DR Learn how to fine-tune the BERT model for text classification. Train and evaluate it on a small dataset for detecting seven intents. The results might surprise you!

Jobs

Projects Publications

comments

By <u>Venelin Valkov</u>, Machine <u>Learning</u> Engineer

Recognizing intent (IR) from text is very useful these days. Usually, you get a short text (sentence or two) and have to classify it into one (or multiple) categories.

Multiple product support systems (help centers) use IR to reduce the need for a large number of employees that copy-and-paste boring responses to frequently asked questions. Chatbots, automated email responders, answer recommenders (from a knowledge base with questions and answers) strive to not let you take the time of a real person.

This guide will show you how to use a pre-trained NLP model that might solve the (technical) support problem that many business owners have. I mean, BERT is freaky good! It is really easy to use, too!

Run the complete notebook in your browser

The complete project on GitHub

Data

The data contains various user queries categorized into seven intents. It is hosted on <u>GitHub</u> and is first presented in <u>this paper</u>.

Here are the intents:

- SearchCreativeWork (e.g. Find me the I, Robot television show)
- GetWeather (e.g. Is it windy ነት ገን ውሳት በተመው ነው። MA right now?)
- BookRestaurant (e.g. I want to book a highly rated restaurant for me and my boyfriend Topics tomorrow night)
 Artificial Intelligence
- PlayMusic (e.g. Play the last track from Beyoncé off Spotify)
- AddToPlaylist (e.g. Add Diamonds to my roadtrip playlist)
 Data Science

Subscribe To Our Newsletter

(Get The Complete Collection of DMa 당한유는은 라마 Sheets & Great Big NLP Primer ebook) NI P Your email address

Search KDnuggets...

Q

Latest Posts

A/B Testing: A Comprehensive Guide

Phi-2: Small LMs that are Doing Big Things

Survey: Machine Learning Projects Still Routinely Fail to Deploy

Enroll in a 4-year Computer Science Degree Program For Free

Data Cleaning in SQL: How To Prepare Messy Data for Analysis

Using Lightning AI Studio For Free

Top Posts

Enroll in a 4-year Computer Science Degree Program For Free

Level 50 Data Scientist: Python Libraries to Know

Prompt Engineering 101: Mastering Effective LLM Communication

Using Lightning AI Studio For Free

Data Cleaning in SQL: How To Prepare Messy Data for Analysis

Run an LLM Locally with LM Studio

What Junior ML Engineers Actually Need to Know to Get Hired?

Too Many Python Versions to Manage? Pyenv to the Rescue

Phi-2: Small LMs that are Doing Big Things

SUBSCRIBE









JOIN NEWSLETTER



!gdown --id 10lcvGWReJMuyYQuOZm149vHWWPtlboR6 --output train.csv !gdown --id 10i5cRlTybuIF2P활ቴያዩ kkqrXrdt77w --output valid.csv !gdown --id 1ep9H6-HvhB4utြduVatizireWNUSG3P_uF --output test.csv

Certificates
Courses
Online Masters

We'll load the data into data frames and expand the training data by merging the training

Resources

and validation intents:

Cheat Sheets

train = pd.read_csv("train.csv")
valid = pd.read_csv("valid.csv")
test = pd.read_csv("test.csv")
Publications

train = train.append(valid).Webenarandex(drop=True)

▼

Get the FREE ebook 'The Great Big Natural Language Processing Primer' and 'The Complete Collection of Data Science Cheat Sheets' along with the leading newsletter on Data Science, Machine Learning, Al & Analytics straight to your inbox.

Your Email

SIGN UP

By subscribing you accept KDnuggets Privacy Policy

We have 13,784 training examples and two columns - text and intent. Let's have a look at the number of texts per intent:



The amount of texts per intent is quite balanced, so we'll not be needing any imbalanced modeling techniques.

BERT

The BERT (Bidirectional Encoder Representations from Transformers) model, introduced in the BERT: Pre-training of Deep Bidirectional Transformers for Language

Understanding paper, made possible achieving State-of-the-art results in a variety of NLP tasks, for the regular ML practitioner. And you can do it without having a large dataset! But how is this possible?

BERT is a pre-trained Transformer Encoder stack. It is trained on Wikipedia and the <u>Book Corpus</u> dataset. It has two versions - Base (12 encoders) and Large (24 encoders).

BERT is built on top of multiple clever ideas by the NLP community. Some examples are <u>ELMo</u>, <u>The Transformer</u>, and the <u>OpenAl Transformer</u>.

ELMo introduced contextual ward embeddings (one word can have a different meaning Top Posts based on the words around it). The straigsformer uses attention mechanisms to understand the context in which the word is being used. That context is then encoded into a vector representation. In practice, it wies a better job with long-term dependencies.

Artificial Intelligence

BERT is a bidirectional model (looks both forward and backward). And the best of all, BERT Computer Vision

can be easily used as a feature partragion of lata. How

Data Science

Subscribe Tb Oวนสริฟิย์พริโอtter

(Get The Complete Collection of DMa 당한 유민이 Sheets & Great Big NLP Primer ebook)
NI P









JOIN NEWSLETTER

Edenity, the dathors of the DETT Paper open sourced their work dions with manapic pro

trained models. The original magazementation is in TensorFlow, but there are very good

<u>PyTorch implementations</u> too!

Certificates

Let's start by downloading one of the simpler pre-trained models and unzip it:

!wget https://storage.googlegpiscegm/bert_models/2018_10_18/uncased_L-12_H-768_A-12.zi !unzip uncased_L-12_H-768_A-12.zi !neets

Events

Jobs

Projects

Publications
This will unzip a checkpoint, confignation vocabulary, along with other files.

Unfortunately, the original implementation is not compatible with TensorFlow 2. The <u>bertfor-tf2</u> package solves this issue.

Preprocessing

We need to convert the raw texts into vectors that we can feed into our model. We'll go through 3 steps:

- Tokenize the text
- Convert the sequence of tokens into numbers
- Pad the sequences so each one has the same length

Let's start by creating the BERT tokenizer:

```
tokenizer = FullTokenizer(
  vocab_file=os.path.join(bert_ckpt_dir, "vocab.txt")
)
```

Let's take it for a spin:

```
tokenizer.tokenize("I can't wait to visit Bulgaria again!")
```

```
['i', 'can', "'", 't', 'wak og 'to', 'visit', 'bulgaria', 'again', '!']

Top Posts

Submissions

About
```

The tokens are in lowercase and the punctuation is available. Next, we'll convert the tokens

Topics to numbers. The tokenizer can Admicial intelligence

Career Advice

tokens = tokenizer.tokenize($\begin{subarray}{cccc} $d_{mparer}(t_{V|Mparer}) $ & tokenizer.convert_tokens_to_ids_{tokens} $ & tokenizer.convert_tokens_to_ids_{tokens} $ & tokenizer.convert_tokens_to_ids_{tokens} $ & tokenizer.convert_tokens_toke$

Data Science

Subscribe To Osure Newsletter

(Get The Complete Collection of DMaching Ceethants heets & Great Big NLP Primer ebook) NLP









JOIN NEWSLETTER

We'll package the preprocessing into a class that is heavily based on the one from this

```
Education
notebook:
                              Certificates
                              Courses
 class IntentDetectionData:
                             Online Masters
   DATA_COLUMN = "text"
   LABEL_COLUMN = "intent"
                            Resources
   def
        __init__(
                              Cheat Sheets
     self,
                              Events
     train,
                             lobs
     test,
     tokenizer: FullTokenizer Publications
                              Projects
     classes,
                             Webinars
     max_seq_len=192
     self.tokenizer = tokenizer
     self.max_seq_len = 0
     self.classes = classes
     ((self.train_x, self.train_y), (self.test_x, self.test_y)) =\
      map(self._prepare, [train, test])
     print("max seq_len", self.max_seq_len)
     self.max_seq_len = min(self.max_seq_len, max_seq_len)
     self.train_x, self.test_x = map(
       self._pad,
       [self.train_x, self.test_x]
     )
   def _prepare(self, df):
     x, y = [], []
     for _, row in tqdm(df.iterrows()):
       text, label =\
        row[IntentDetectionData.DATA_COLUMN], \
        row[IntentDetectionData.LABEL_COLUMN]
       tokens = self.tokenizer.tokenize(text)
       tokens = ["[CLS]"] + tokens + ["[SEP]"]
       token_ids = self.tokenizer.convert_tokens_to_ids(tokens)
       self.max_seq_len = max(self.max_seq_len, len(token_ids))
       x.append(token_ids)
       y.append(self.classes.index(label))
     return np.array(x), np.array(y)
   def _pad(self, ids):
     x = []
     for input_ids in ids:
       input_ids = input_ids[:min(len(input_ids), self.max_seq_len - 2)]
       input_ids = input_ids + [0] * (self.max_seq_len - len(input_ids))
       x.append(np.array(input_ids))
     return np.array(x)
                            Blog
                              Top Posts
We figure out the padding length by taking the minimum between the longest text and the
                              About
max sequence length parameter. We also surround the tokens for each text with two
special tokens: start with [CL'S
                              and end with [SEP].
                              Career Advice
```

Subscribe To Our Newsletter

Fine-tuning

Computer Vision Data Engineering

Data Science

(Get The Complete Collection of DMa 상ireh는은 fields heets & Great Big NLP MLOps Primer ebook) NLP









in

JOIN NEWSLETTER

```
with tf.io.gfile.GFile(bert_config_file, "r") as reader:
    bc = StockBertConfig_fromeTson_string(reader.read())
    bert_params = map_stb@kl_@biOffig_to_params(bc)
    bert_params.adapter_sizertificNose
    bert = BertModelLayer.foomeparams(bert_params, name="bert")
input_ids = keras.layers.Input(
    shape=(may cos ?)
  shape=(max_seq_len, ), Resources
  dtype='int32',
                              Cheat Sheets
  name="input_ids"
                              Events
bert_output = bert(input_ids)
print("bert shape", bert_output.shape)
Projects
Projects
Publications
cls_out = keras.layers.Lambda(Tambda seq: seq[:, 0, :])(bert_output)
cls_out = keras.layers.Dropout(0.5)(cls_out)
logits = keras.layers.Dense(units=768, activation="tanh")(cls_out)
logits = keras.layers.Dropout(0.5)(logits)
logits = keras.layers.Dense(
  units=len(classes),
  activation="softmax"
)(logits)
model = keras.Model(inputs=input_ids, outputs=logits)
model.build(input_shape=(None, max_seq_len))
load_stock_weights(bert, bert_ckpt_file)
return model
```

We're fine-tuning the pre-trained BERT model using our inputs (text and intent). We also flatten the output and add Dropout with two Fully-Connected layers. The last layer has a softmax activation function. The number of outputs is equal to the number of intents we have - seven.

You can now use BERT to recognize intents!

Training

It is time to put everything together. We'll start by creating the data object:

```
classes = train.intent.unique().tolist()
 data = IntentDetectionDataglog
   train,
                               Top Posts
   test,
                               Submissions
   tokenizer,
                               About
   classes,
   max_seq_len=128
                              Topics
                               Artificial Intelligence
                               Career Advice
                                Computer Vision
We can now create the model using the maximum sequence length:
                                Data Science
```

Subscribe To Our Newsletter

(Get The Complete Collection of DMa Stire Le 라마이 Sheets & Great Big NLP MLOps Primer ebook) NLP









in

JOIN NEWSLETTER

Datasets

Education You'll notice that even this "slim" BERT has almost 110 million parameters. Indeed, your

model is HUGE (that's what she ওধাৰ).

Online Masters

Fine-tuning models like BERT so both art and doing tons of failed experiments. Fortunately, Resources

the authors made some recommendations:

Events
Jobs
Projects

Learning rate (Adam): 5e-5, Palis, tiges Webinars

Number of epochs: 2, 3, 4

```
model.compile(
  optimizer=keras.optimizers.Adam(1e-5),
  loss=keras.losses.SparseCategoricalCrossentropy(from_logits=True),
  metrics=[keras.metrics.SparseCategoricalAccuracy(name="acc")]
)
```

We'll use Adam with a slightly different learning rate (cause we're badasses) and use sparse categorical crossentropy, so we don't have to one-hot encode our labels.

Let's fit the model:

```
log_dir = "log/intent_detection/" +\
  datetime.datetime.now().strftime("%Y%m%d-%H%M%s")
tensorboard_callback = keras.callbacks.TensorBoard(log_dir=log_dir)

model.fit(
  x=data.train_x,
  y=data.train_y,
  validation_split=0.1,
  batch_size=16,
  shuffle=True,
  epochs=5,
  callbacks=[tensorboard_callback]
)
```

We store the training logs, so you can explore the training process in <u>Tensorboard</u>. Let's

have a look:

Blog
Top Posts
Submissions
About
▼
Topics
Artificial Intelligence
Career Advice
Computer Vision
Data Engineering
Data Science

Subscribe To Osur Newsletter

(Get The Complete Collection of DMa Stire Le 라마이 Sheets & Great Big NLP MLOps Primer ebook) NLP









in

JOIN NEWSLETTER

```
I got to be honest with you. Datasets Was impressed with the results. Training using only 12.5k samples we got:

Certificates
Courses

_, train_acc = model.evaluate(linet Mathaterin x, data.train_y)
_, test_acc = model.evaluate(data.test_x, data.test_y)

Resources
print("train acc", train_acc)heat Sheets
print("test acc", test_acc)

Vents

Jobs
Projects
Publications

train acc 0.9915119
test acc 0.9771429

Webinars
```

Impressive, right? Let's have a look at the confusion matrix:

Finally, let's use the model to detect intent from some custom sentences:

```
sentences = [
   "Play our song now",
   "Rate this book as awful"
 pred_tokens = map(tokenizer.tokenize, sentences)
 \label{eq:pred_tokens} \verb| pred_tokens = map(lambda tok: ["[CLS]"] + tok + ["[SEP]"], pred_tokens)| \\
 pred_token_ids = list(map(tokenizer.convert_tokens_to_ids, pred_tokens))
 pred_token_ids = map(
   lambda tids: tids +[0]*(data.max_seq_len-len(tids)),
   pred_token_ids
 pred_token_ids = np.array(list(pred_token_ids))
 predictions = model.predict(pred_token_ids).argmax(axis=-1)
 for text, label in zip(sentences, predictions):
   print("text:", text, "\nintent:", classes[label])
   print()
                               Blog
                                 Top Posts
                                 Submissions
 text: Play our song now
                                 About
 intent: PlayMusic
 text: Rate this book as awforbics
 intent: RateBook
                                 Artificial Intelligence
                                 Career Advice
                                 Computer Vision
                                 Data Engineering
Man, that's (clearly) gangsta! Okathe examples might not be as diverse as real queries might
```

Subscribe To Our Newsletter

(Get The Complete Collection of DMacstine Le은 유현화용 heets & Great Big NLP MLOps Primer ebook) NLP









JOIN NEWSLETTER

DENT MODEL FOR CONCINCUITOR, FOR PRODUCTY AIRCRAY

know that you can use it for parasiety of other tasks, too! You just have to fiddle with the

Education layers. EASY! Certificates

Courses Run the complete notebook in your browser

The complete project on Gittlybrces

Cheat Sheets

Doing Al/ML feels a lot like having superpowers, right? Thanks to the wonderful NLP

community, you can have superpowers, too! What will you use them for?

Publications Webinars

References

- BERT Fine-Tuning Tutorial with PyTorch
- SNIPS dataset
- The Illustrated BERT, ELMo, and co.
- BERT for dummies Step by Step Tutorial
- Multi-label Text Classification using BERT The Mighty Transformer
- Deep Learning
- Keras
- NLP
- Text Classification
- Python

Continue Your Machine Learning Journey:

Blog

Top Posts

Submissions

About

Topics

Hacker's Guide to Washine Learning with Python

Career Advice

A hands-on guide to solving of aphter or Id. Machine Learning problems with Scikit-Learn,

Data Engineering TensorFlow 2, and Keras Data Science

Subscribe To Our Newsletter

(Get The Complete Collection of Data Science Cheat Sheets & Great Big NLP MLOps Primer ebook) NLP









JOIN NEWSLETTER

Datasets
Education
Certificates
Courses

Online Masters

Hands-On Machine Learning from Scratch

Resources

Develop a deeper understanding notels, tools and concepts by

building them from scratch with Python

Projects

Publications BUY THE BOOK

Webinars

•

Deep Learning for JavaScript Hackers

Beginners guide to understanding Machine Learning in the browser with TensorFlow.js

BUY THE BOOK

Bio: <u>Venelin Valkov</u> (<u>GitHub</u>) is a Machine Learning Engineer working on document data extraction using Deep Learning. In his free time, he likes to write, work on side projects, ride his bike, and do deadlifts!

Original. Reposted with permission.

Related:

- BERT is changing the NLP landscape
- Lit BERT: NLP Transfer Learning In 3 Steps
- BERT, RoBERTa, DistilBERT &LNet: Which one to use?

Top Posts Submissions

More On This Topic

About

- Building and Training Your First Neural Network with TensorFlow and Keras
 Topics
- How to Train a Joint Entities and Relation Extraction Classifier...
- <u>Building a Knowledge Grap ମଫି ମଣ୍ଡି ଓଡ଼ିଶ</u>rch Using BERT Data Engineering
- Classifying Long Text Documents গুলার BERT

Subscribe Tb Oวนสิริฟิต์พริโอtter

(Get The Complete Collection of DMa 당한 유민 문학자 (Get The Complete Collection of DMa 당한 유민 문학자 (Get The Complete Sheets & Great Big NLP Primer ebook) NLP NLP









JOIN NEWSLETTER

Datasets

Get the Great Big Natural Language Processing Primer and 'The Complete Collection of Data Science Cheat Sheets' along with the leading newsletterling Matterscience, Machine Learning, AI &

Analytics straight to your inbox. Resources

Cheat Sheets

Events Your Emails

Projects

Publications SIGN UP

Webinars

By subscribing you accept KDnuggets Privacy Policy

Next post => <= Previous post

© 2024 Guiding Tech Media | About | Contact | Privacy Policy | Terms of Service

Subscribe To Our Newsletter

(Get The Complete Collection of Data Science Cheat Sheets & Great Big NLP Primer ebook)

