Coursera — Transfer Learning for NLP with TensorFlow Hub

aakashgoel12.medium.com/coursera-transfer-learning-for-nlp-with-tensorflow-hub-7bd494886b87

April 22, 2021

Aakash Goel

·Just now

Objectives → By end of this, you will be able to use

- Various Pre-trained NLP Models from TensorFlow Hub (thub.dev)
- Use transfer learning to fine tune text data
- Visualize Model Performance metric with Tensorboard

Image 01: Example of using pre-trained model in Keras

Problem Statement → Detect Toxic Comment (Quora Insincere Questions Classification)

!nvidia-smi → Command to tell GPU status

```
!nvidia-smi
```

```
Sat Apr 17 12:11:54 2021
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 460.67       Driver Version: 460.32.03    CUDA Version: 11.2     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|                               |                      |               MIG M. |
|===============================+======================+======================|
|   0  Tesla T4            Off  | 00000000:00:04.0 Off |                    0 |
| N/A   49C    P8    10W /  70W |      0MiB / 15109MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                                  |
|  GPU   GI   CI        PID   Type   Process name                  GPU Memory |
|        ID   ID                                                   Usage      |
|=============================================================================|
|  No running processes found                                                 |
+-----------------------------------------------------------------------------+
```

https://colab.research.google.com/drive/1056qVx2GE7QRog_uuC8LqPQqIW-cdK2q?authuser=3#scrollTo=PwWXcwn04wB9

Notebook on colab is opened using A/C → goelaakash12@gmail.com

```
import tensorflow as tf
print("Version: ", tf.__version__)
print("Hub version: ", hub.__version__)
print("GPU is", "available" if tf.config.list_physical_devices('GPU') else "NOT AVAILABLE")
```

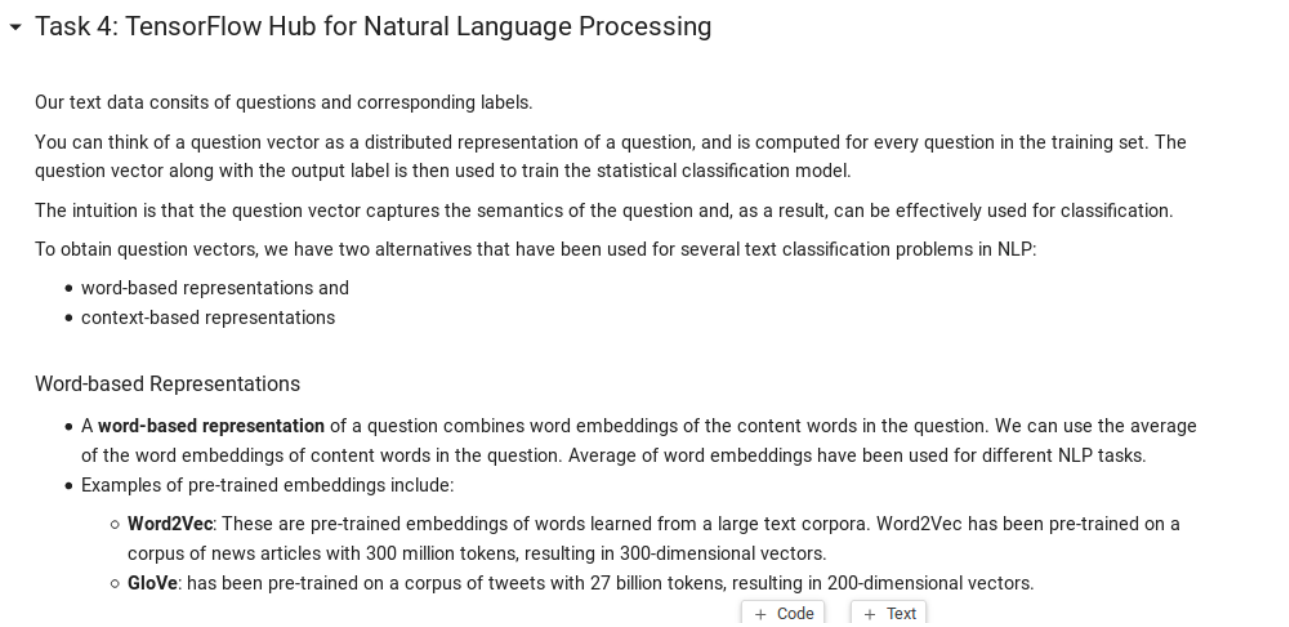


Image 02: Sentence Representation

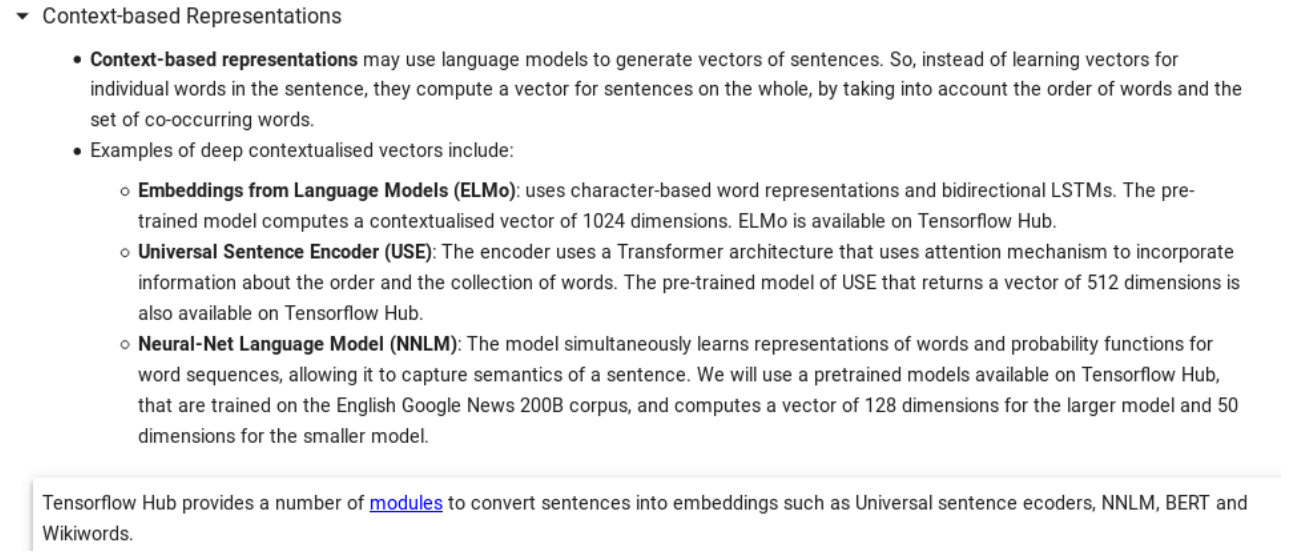


Image 03: Sentence Representation

TensorFlow Hub

Edit description

tfhub.dev

SWIVEL — EMBEDDING (20 DIMENSION)

```
import tensorflow_hub as hub

embed = hub.load("https://tfhub.dev/google/tf2-preview/gnews-swivel-20dim/1")
embeddings = embed(["cat is on the mat", "dog is in the fog"])
```

It can also be used within Keras:

```
hub_layer = hub.KerasLayer("https://tfhub.dev/google/tf2-preview/gnews-swivel-20dim/1", output_shape=[20],
                           input_shape=[], dtype=tf.string)

model = keras.Sequential()
model.add(hub_layer)
model.add(keras.layers.Dense(16, activation='relu'))
model.add(keras.layers.Dense(1, activation='sigmoid'))
```

```
[12] module_url = "https://tfhub.dev/google/tf2-preview/gnews-swivel-2
```

module_url: https://tfhub.dev/google/tf2-preview/gnews-swivel-20dim

Tasks 5 & 6: Define Function to Build and Compile Models

```
def train_and_evaluate_model(module_url, embed_size, name, trainable=False):
  hub_layer = hub.KerasLayer(module_url, input_shape=[], output_shape=[embed_size], dtype=tf.string, trainable=trainable)
  model = tf.keras.models.Sequential([
                                      hub_layer,
                                      tf.keras.layers.Dense(256, activation='relu'),
                                      tf.keras.layers.Dense(64, activation='relu'),
                                      tf.keras.layers.Dense(1, activation='sigmoid')
  ])
```

```
def train_and_evaluate_model(module_url, embed_size, name, trainable=False):
  hub_layer = hub.KerasLayer(module_url, input_shape=[], output_shape=[embed_size], dtype=tf.string, trainable=trainable)
  model = tf.keras.models.Sequential([
                                      hub_layer,
                                      tf.keras.layers.Dense(256, activation='relu'),
                                      tf.keras.layers.Dense(64, activation='relu'),
                                      tf.keras.layers.Dense(1, activation='sigmoid')
  ])
  model.compile(optimizer=tf.keras.optimizers.Adam(learning_rate=0.0001),
                loss=tf.losses.BinaryCrossentropy(),
                metrics=[tf.metrics.BinaryAccuracy(name='accuracy')])
  history = model.fit(train_df['question_text'], train_df['target'],
                      epochs=100,
                      batch_size=32,
                      validation_data=(valid_df['question_text'], valid_df['target']),
                      callbacks=[tfdocs.modeling.EpochDots(),
                                 tf.keras.callbacks.EarlyStopping(monitor='val_loss', patience=2, mode='min'),
                                 tf.keras.callbacks.TensorBoard(logdir/name)],
                      verbose=0)
```

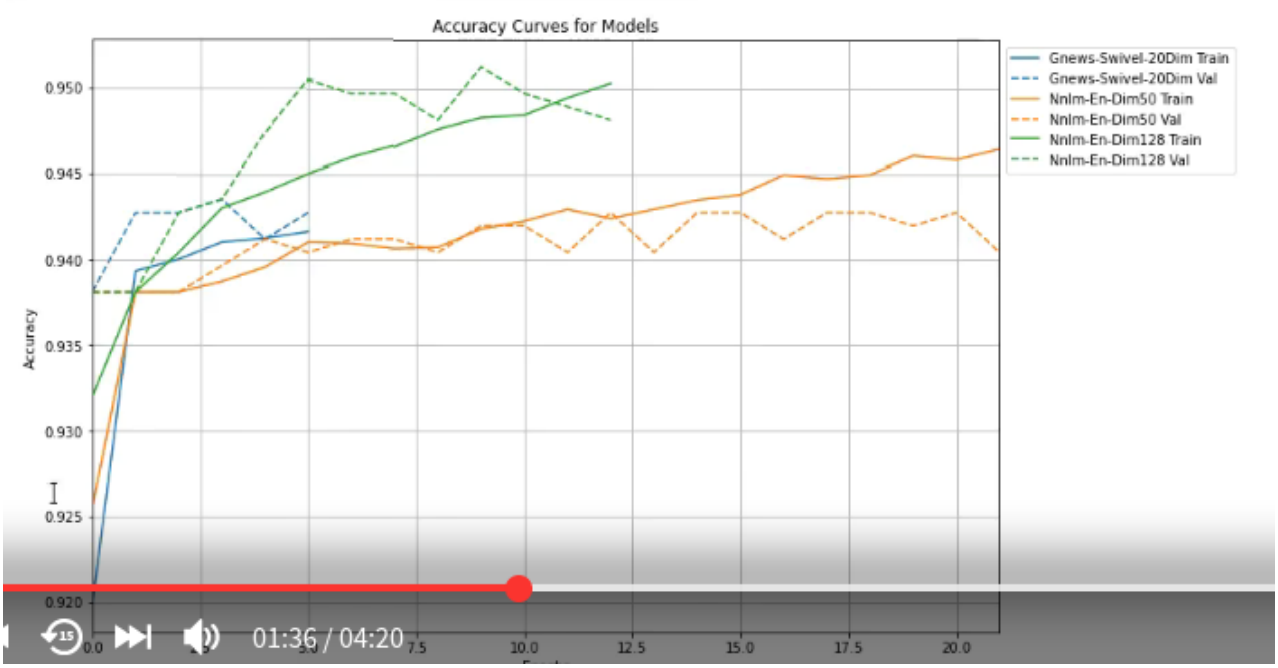
Task 8: Compare Accuracy and Loss Curves

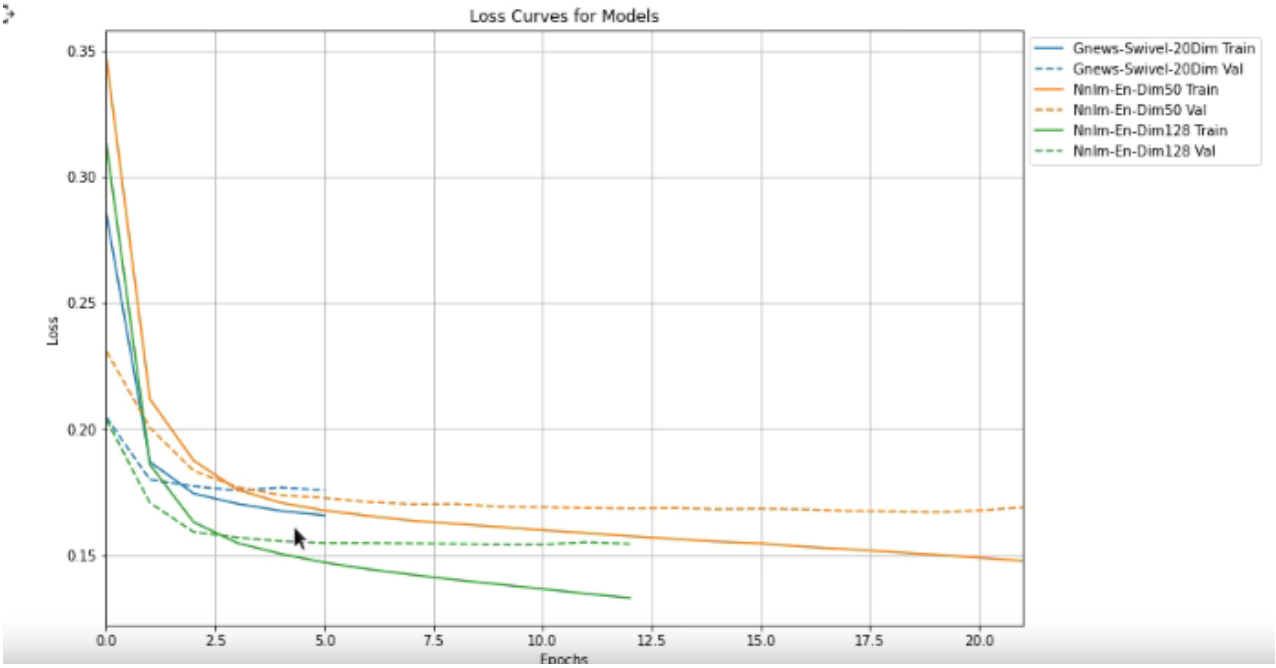
```
plt.rcParams['figure.figsize'] = (12, 8)
plotter = tfdocs.plots.HistoryPlotter(metric = 'accuracy')
plotter.plot(histories)
plt.xlabel("Epochs")
plt.legend(bbox_to_anchor=(1.0, 1.0), loc='upper left')
plt.title("Accuracy Curves for Models")
plt.show()
```

Task 8: Compare Accuracy and Loss Curves

```
plt.rcParams['figure.figsize'] = (12, 8)
plotter = tfdocs.plots.HistoryPlotter(metric = 'accuracy')
plotter.plot(histories)
plt.xlabel("Epochs")
plt.legend(bbox_to_anchor=(1.0, 1.0), loc='upper left')
plt.title("Accuracy Curves for Models")
plt.show()
```

```
[ ] plotter = tfdocs.plots.HistoryPlotter(metric = 'loss')
    plotter.plot(histories)
    plt.xlabel("Epochs")
    plt.legend(bbox_to_anchor=(1.0, 1.0), loc='upper left')
    plt.title("Loss Curves for Models")
    plt.show()
```





Quiz

1. Which of the following models generate context-based representations of text? (Select all that apply)

- [x] Embeddings from Language Models (ELMo)

> ✓ **Correct**
>
> Correct! ELMo uses character-based word representations and bidirectional LSTMs. The pre-trained model computes a contextualised vector of 1024 dimensions.

- [ ] GloVe

- [x] Universal Sentence Encoder (USE)

> ✓ **Correct**
>
> Correct. USE uses a Transformer architecture that uses an attention mechanism to incorporate information about the order and the collection of words. The pre-trained model of USE that returns a vector of 512 dimensions is also available on Tensorflow Hub.

- [x] Neural-Net Language Model (NNLM)

> ✓ **Correct**
>
> You got it! The model simultaneously learns representations of words and probability functions for word sequences, allowing it to capture semantics of a sentence.
>
> In the project, we used pretrained models available on Tensorflow Hub, that are trained on the English Google News 200B corpus, and computed vectors of 128 dimensions for the larger model and 50 dimensions for the smaller model.

2. In the hands-on project, did we write code to preprocess the text? Please explain. Preprocessing steps can include removing stop words, stemming, lemmatization, tf-idf, tokenization, padding, etc.

- ( ) Yes. The text corpus needed to be appropriately vectorized (converted into a numerical representation) for the classification models to be able to use them. Theorefore, we used text preprocessing modules from **tf.keras** to turn each text into a sequence of integers.

- (•) No. The text embedding modules we used from TF Hub preprocess the the input text. The modules do not require preprocessing the data before applying the modules, as preprocessing of input text is part of the TensorFlow graph.

> ✓ **Correct**
>
> Good job! For more, read the preprocessing section for the Universal Sentence Encoder and NNLM over on TF Hub.

3. Say you decide to build a binary text classification model to identify positive and negative sentiments associated with product reviews on Amazon. As you've taken this project, you have learned about using TensorFlow Hub and it's pre-traiined text embedding modules for NLP. How would you load TF Hub modules in your sequential module?

```
hub_layer = hub.KerasLayer("https://tfhub.dev/google/tf2-preview/nnlm-en-dim12
                           input_shape=[], dtype=tf.string)

model = keras.Sequential()
model.add(hub_layer)
model.add(keras.layers.Dense(16, activation='relu'))
model.add(keras.layers.Dense(1, activation='sigmoid'))

model.summary()
```

```
hub_layer = hub.load("https://tfhub.dev/google/tf2-preview/nnlm-en-dim128/1",
                           input_shape=[], dtype=tf.string)

model = keras.Sequential()
model.add(hub_layer)
model.add(keras.layers.Dense(16, activation='relu'))
model.add(keras.layers.Dense(1, activation='sigmoid'))

model.summary()
```

```
model = keras.Sequential()
model.add("https://tfhub.dev/google/tf2-preview/nnlm-en-dim128/1", output_shap
model.add(keras.layers.Dense(16, activation='relu'))
model.add(keras.layers.Dense(1, activation='sigmoid'))
```

4. While visualizing your model's accuracy with Tensorboard, you obtain the following graph. The vertical axis represent model accuracy, while the horizontal axis represents the number of training epochs. The training accuracy and validation accuracy are given by the blue and red curves, respectively.

   What do you think the model is doing?

   - ○ Underfitting
   - ○ Coverged to a global optima
   - ◉ Overfitting

   ✓ **Correct**

   Correct! We see that the accuracy of our model on the validation data would peaks after training for a number of epochs, and then stagnates and starts decreasing. In other words, our model overfit to the training data. Learning how to deal with overfitting is important. Although it's often possible to achieve high accuracy on the training set, what we really want is to develop models that generalize well to a testing set (or data they haven't seen before).

5. Why should transfer learning work in NLP? Feel free to revisit the recordings and notebook. What are your intuitions, if any?

   Semantic Representation of text learned in Pre-trained model (trained on large corpus) and while solving NLP problem, we can use pre-trained model which gives state of art accuracy in lot of NLP task. Also, when training data is very less, we can use pre-trained model.

   ✓ **Correct**

   Here are a few reasons I could think of:

   - Many NLP tasks share common knowledge about language (linguistic representations, structural similarities, syntax, semantics...).
   - Annotated data is rare, make use of as much supervision as possible. If you can combine data sets that you used for several tasks to get much bigger datasets. Bigger datasets are generally better for deep learning models.
   - Unlabelled data is abundant (e.g. on the world wide web) and one should try to use as much of it as possible.
   - Empirically, transfer learning has resulted in SOTA results for many supervised NLP tasks (e.g. classification, information extraction, Q&A, etc).

END of QUIZ

More from Aakash Goel

More From Medium

Train on Cloud GPUs with Azure Machine Learning SDK for Python.

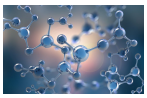
Ben Bogart in Towards Data Science

Training a Computer Vision 101

L.A. Randrup

SMILES Toxicity Prediction

2040 Final Project



How a data scientist solved the Kaggle Titanic competition question without using an estimator

Tracyrenee in CodeX

How to Kaggle the Engineer way. Act 2—Google Colab

Vitalii Kozhukhivskyi in Towards Data Science

OpenAI GPT leaking your data

Mastafa Foufa in Towards Data Science

Boosting Image Captioning Model Performance

Griffin McCauley in Analytics Vidhya

About

Help

Legal

Medium