Why Tensorflow Hub

arXiv:1810.04805v2 [cs.CL] 24 May 2019

BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

Jacob Devlin Ming-Wei Chang Kenton Lee Kristina Toutanova
Google AI Language
{jacobdevlin,mingweichang,kentonl,kristout}@google.com

Abstract

We introduce a new language representation model called **BERT**, which stands for **B**idirectional **E**ncoder **R**epresentations from **T**ransformers. Unlike recent language representation models (Peters et al., 2018a; Radford et al., 2018), BERT is designed to pre-train deep bidirectional representations from unlabeled text by jointly conditioning on both left and right context in all layers. As a result, the pre-trained BERT model can be fine-tuned with just one additional output layer to create state-of-the-art models for a wide range of tasks, such as question answering and language inference, without substantial task-specific architecture modifications.

BERT is conceptually simple and empirically powerful. It obtains new state-of-the-art results on eleven natural language processing tasks, including pushing the GLUE score to 80.5% (7.7% point absolute improvement), MultiNLI accuracy to 86.7% (4.6% absolute improvement), SQuAD v1.1 question answering Test F1 to 93.2 (1.5 point absolute improvement) and SQuAD v2.0 Test F1 to 83.1 (5.1 point absolute improvement).

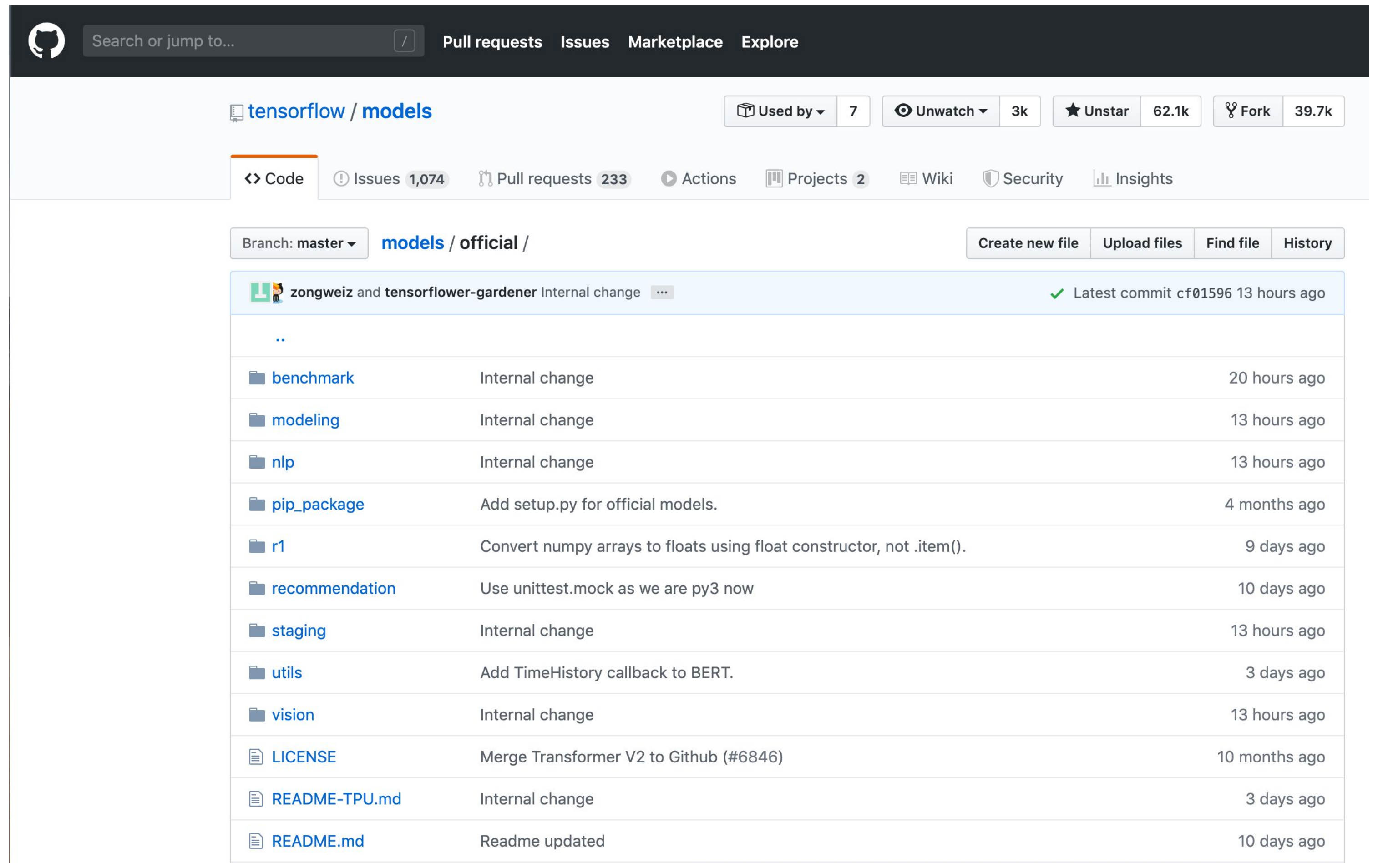
1 Introduction

Language model pre-training has been shown to be effective for improving many natural language processing tasks (Dai and Le, 2015; Peters et al., 2018a; Radford et al., 2018; Howard and Ruder, 2018). These include sentence-level tasks such as natural language inference (Bowman et al., 2015; Williams et al., 2018) and paraphrasing (Dolan and Brockett, 2005), which aim to predict the relationships between sentences by analyzing them holistically, as well as token-level tasks such as named entity recognition and question answering, where models are required to produce fine-grained output at the token level (Tjong Kim Sang and De Meulder, 2003; Rajpurkar et al., 2016).

There are two existing strategies for applying pre-trained language representations to downstream tasks: *feature-based* and *fine-tuning*. The feature-based approach, such as ELMo (Peters et al., 2018a), uses task-specific architectures that include the pre-trained representations as additional features. The fine-tuning approach, such as the Generative Pre-trained Transformer (OpenAI GPT) (Radford et al., 2018), introduces minimal task-specific parameters, and is trained on the downstream tasks by simply fine-tuning *all* pre-trained parameters. The two approaches share the same objective function during pre-training, where they use unidirectional language models to learn general language representations.

We argue that current techniques restrict the power of the pre-trained representations, especially for the fine-tuning approaches. The major limitation is that standard language models are unidirectional, and this limits the choice of architectures that can be used during pre-training. For example, in OpenAI GPT, the authors use a left-to-right architecture, where every token can only attend to previous tokens in the self-attention layers of the Transformer (Vaswani et al., 2017). Such restrictions are sub-optimal for sentence-level tasks, and could be very harmful when applying fine-tuning based approaches to token-level tasks such as question answering, where it is crucial to incorporate context from both directions.

In this paper, we improve the fine-tuning based approaches by proposing BERT: **B**idirectional **E**ncoder **R**epresentations from **T**ransformers. BERT alleviates the previously mentioned unidirectionality constraint by using a "masked language model" (MLM) pre-training objective, inspired by the Cloze task (Taylor, 1953). The masked language model randomly masks some of the tokens from the input, and the objective is to predict the original vocabulary id of the masked



How do I use it?

Is it safe?

Is it fair?

Is it the latest version?

tfhub.dev

TensorFlow Hub

A comprehensive collection of models

Image

Text

Video

Audio

Before starting

Pre-trained models ready for transfer learning on your own datasets and deployable anywhere you want

TensorFlow Extended

TensorFlow Lite

TensorFlow.JS

Coral

Filters Clear all

- Problem domain
- Saved model format
- Architecture
- Publisher
- Dataset
- Language

Text embedding
nnlm-en-dim128-with-normalization
Published by: Google Updated: 10/24/2019
Token based text embedding trained on English Google News 200B corpus.
NNLM | Google News

Collection
universal-sentence-encoder
Published by: Google Updated: 10/24/2019
Collection of universal sentence encoders trained on variety of data.
DAN,Transformer

Image pose detection .JS
posenet/mobilenet/quantized/1/100
Published by: TensorFlow Updated: 10/24/2019
PoseNet model for pose esimation.

Image classification
imagenet/mobilenet_v1_025_224...
Published by: Google Updated: 10/24/2019
Imagenet (ILSVRC-2012-CLS) classification with MobileNet V1 (depth multiplier 0.25).
MobileNet V1 | ImageNet (ILSVRC-201...

Image classification .JS
imagenet/mobilenet_v2_075_224...
Published by: Google Updated: 10/24/2019
Imagenet (ILSVRC-2012-CLS) classification with MobileNet V2 (depth multiplier 0.75).
MobileNet V2 | ImageNet (ILSVRC-201...

Image classification .JS
imagenet/mobilenet_v1_025_128...
Published by: Google Updated: 10/24/2019
Imagenet (ILSVRC-2012-CLS) classification with MobileNet V1 (depth multiplier 0.25).
MobileNet V1 | ImageNet (ILSVRC-201...

Image segmentation .JS
bodypix_050
Published by: TensorFlow Updated: 10/24/2019
A person segmentation model.

Other
llr-pretrain-adv/latents
Published by: DeepMind Updated: 10/24/2019
LLR-pretrained ResNet-152 on ImageNet then adversarially trained for the unrestricted adversarial
Other | Other

Image generator
compare_gan/s3gan_20_128x128
Published by: Google Updated: 10/24/2019
S3GAN trainend on ImageNet with 20% labels.
Other | ImageNet (ILSVRC-201...

Image pose detection .JS
posenet/mobilenet/quantized/2/050
Published by: TensorFlow Updated: 10/24/2019
PoseNet model for pose esimation.

Text embedding
universal-sentence-encoder-xling/en-fr
Published by: Google Updated: 10/24/2019
English and French language-agnostic text encoder.
Transformer

Publisher
tensorflow
TensorFlow

Image classification
imagenet/pnasnet_large/classification
Published by: Google Updated: 10/24/2019
Imagenet (ILSVRC-2012-CLS) classification with PNASNet-5 (large).
PNASNet-5 (large) | ImageNet (ILSVRC-201...

Image feature vector .JS
imagenet/mobilenet_v1_025_192...
Published by: Google Updated: 10/24/2019
Feature vectors of images with MobileNet V1 (depth multiplier 0.25) trained on ImageNet (ILSVRC-2012-
MobileNet V1 | ImageNet (ILSVRC-201...

Image generator
biggan-deep-256
Published by: DeepMind Updated: 10/24/2019
BigGAN-deep image generator trained on 256x256 ImageNet.
Other | ImageNet (ILSVRC-201...

Image pose detection .JS
posenet/mobilenet/float/050
Published by: TensorFlow Updated: 10/24/2019
PoseNet model for pose esimation.

Image feature vector
vae
Published by: Vtab Updated: 10/24/2019
Visual representation obtained by training a VAE on ImageNet.
Other | ImageNet (ILSVRC-201...

Image generator
compare_gan...
Published by: Google Updated: 10/24/2019
ResNet19 trained on CelebA HQ (128x128) (FID: 35.85).
Other | CelebA HQ

Text embedding
bert_en_cased_L-24_H-1024_A-16
Published by: TensorFlow Updated: 10/24/2019
Bidirectional Encoder Representations from Transformers (BERT).
Transformer | Wikipedia and BooksC...

Text embedding
tf2-preview/nnlm-de-dim50-with-...
Published by: Google Updated: 10/24/2019
Token based text embedding trained on German Google News 30B corpus.
NNLM | Google News

Image feature vector .JS
imagenet/mobilenet_v2_100_160...
Published by: Google Updated: 10/24/2019
Feature vectors of images with MobileNet V2 (depth multiplier 1.00) trained on ImageNet (ILSVRC-2012-
MobileNet V2 | ImageNet (ILSVRC-201...

Text embedding
nnlm-ja-dim128-with-normalization
Published by: Google Updated: 10/24/2019
Token based text embedding trained on Japanese Google News 6B corpus.
NNLM | Google News

Image classification .JS
imagenet/mobilenet_v2_100_224...
Published by: Google Updated: 10/24/2019
Imagenet (ILSVRC-2012-CLS) classification with MobileNet V2 (depth multiplier 1.00).
MobileNet V2 | ImageNet (ILSVRC-201...

Image classification
unsupervised-adversarial-training...
Published by: DeepMind Updated: 10/24/2019
UAT++ adversarially trained WRN-106 (wide residual network) model using 80m@200K unlabeled data and
Other | CIFAR-10

Image feature vector
imagenet/resnet_v2_152/feature_vector
Published by: Google Updated: 10/24/2019
Feature vectors of images with ResNet V2 152 trained on ImageNet (ILSVRC-2012-CLS).
ResNet V2 152 | ImageNet (ILSVRC-201...

Image generator
biggan-deep-512
Published by: DeepMind Updated: 10/24/2019

Image feature vector
rotation
Published by: Vtab Updated: 10/24/2019

Image feature vector
imagenet/mobilenet_v1_100_128...
Published by: Google Updated: 10/24/2019

Image classification TFLite
mobilenet_v1_0.25_224
Published by: TensorFlow Updated: 10/24/2019

Image classification TFLite
mobilenet_v1_0.75_192
Published by: TensorFlow Updated: 10/24/2019

← imagenet/mobilenet_v2_050_96/feature_vector

Problem domain

Image feature vector

Architecture

MobileNet V2

Publisher

Google

Dataset

ImageNet (ILSVRC-2012-CLS)

Format: TF2.0 Saved Model

Fine tunable: Yes

License: Apache-2.0

Last updated: 2020-02-20

Model formats

Saved Model | .JS (v1, default) | .JS (v2, default) | .JS (v3, default)

Want to use this model?

To use this model, take a look at the example code, or at our user guide.

You can also try out the associated Colab.

Copy URL to clipboard

Download Model

Open Colab Notebook

Asset size: 2.62MB

TF2 SavedModel

This is a SavedModel in TensorFlow 2 format. Using it requires TensorFlow 2 (or 1.15) and TensorFlow Hub 0.5.0 or newer.

Overview

MobileNet V2 is a family of neural network architectures for efficient on-device image classification and related tasks, originally published by

- Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, Liang-Chieh Chen: "Inverted Residuals and Linear Bottlenecks: Mobile Networks for Classification, Detection and Segmentation", 2018.

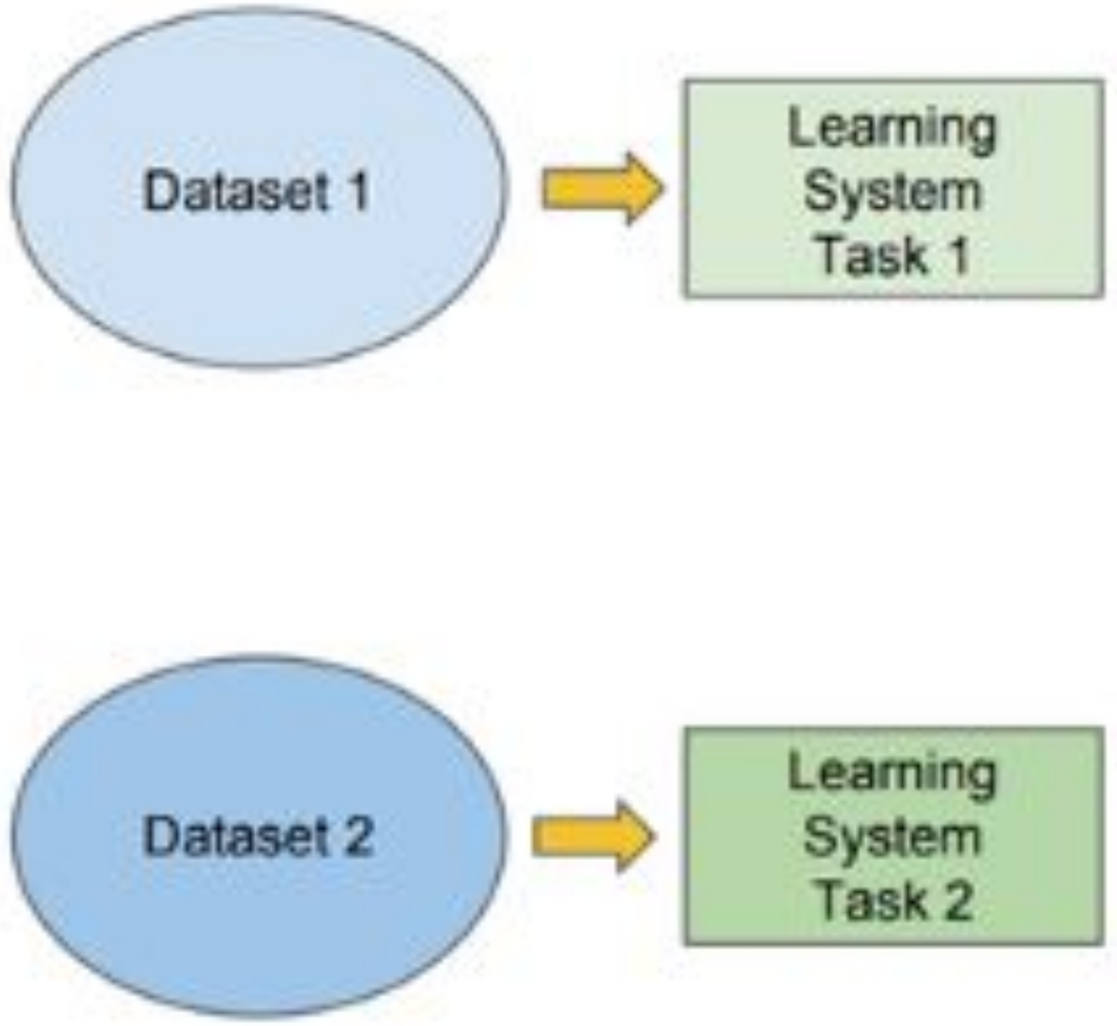
Mobilenets come in various sizes controlled by a multiplier for the depth (number of features) in the convolutional layers. They can also be trained for various sizes of input images to control inference speed.

Tansfer Learning

Traditional ML vs Transfer Learning

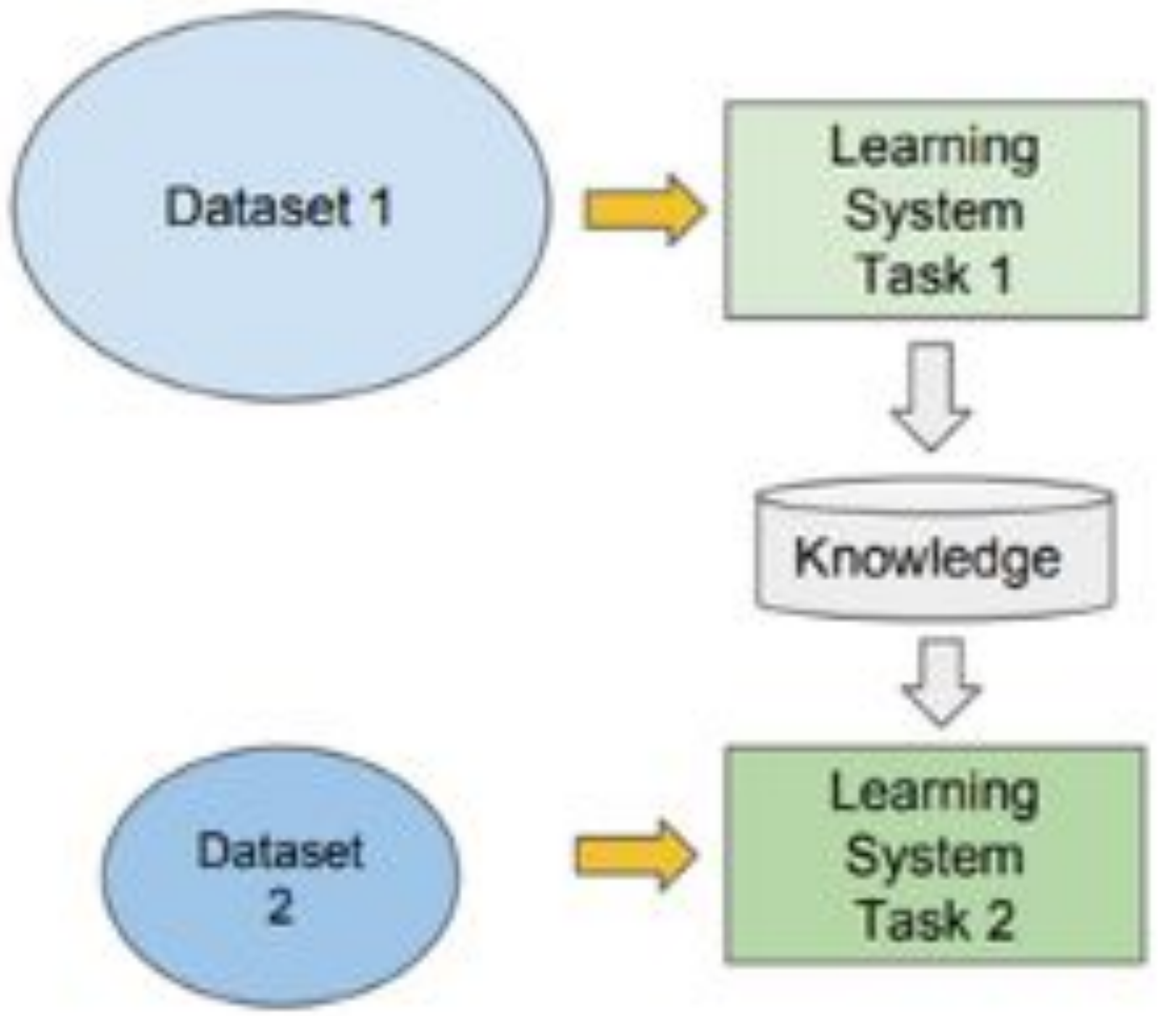
Traditional ML

- Isolated, single task learning:
  - Knowledge is not retained or accumulated. Learning is performed w.o. considering past learned knowledge in other tasks



Transfer Learning

- Learning of a new tasks relies on the previous learned tasks:
  - Learning process can be faster, more accurate and/or need less training data

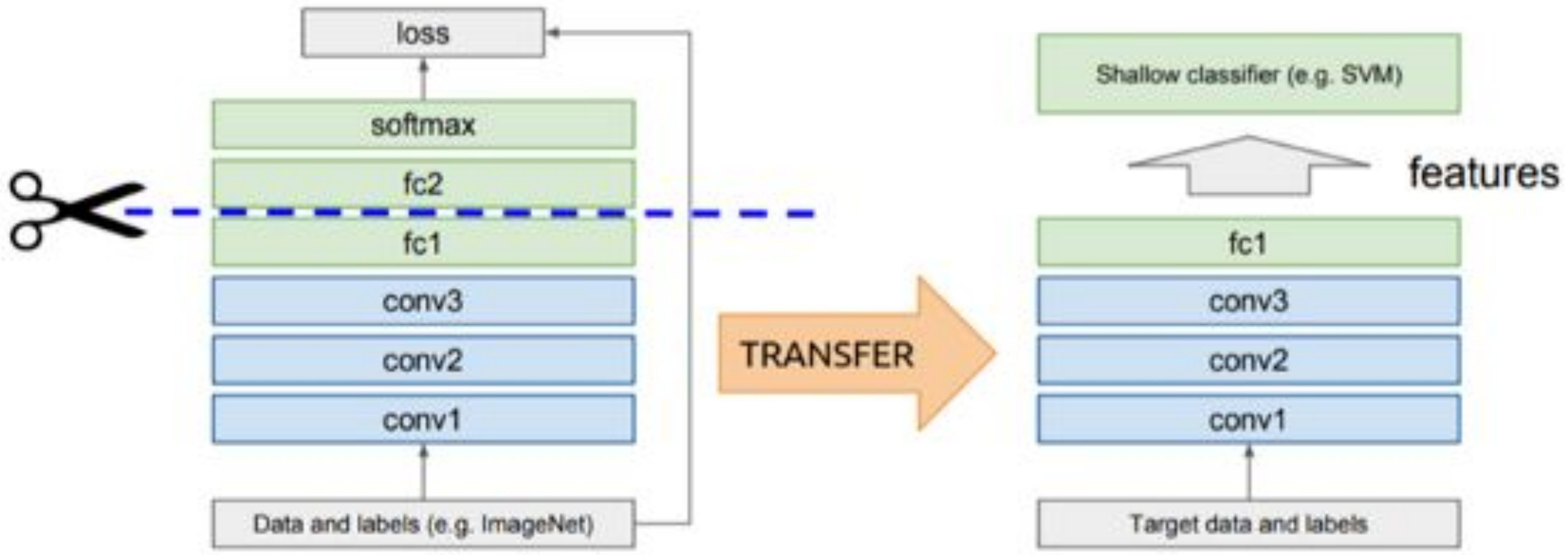


https://miro.medium.com/max/700/1*qfQ3hmHLwApXZBN-A85r8g.png

Tansfer Learning

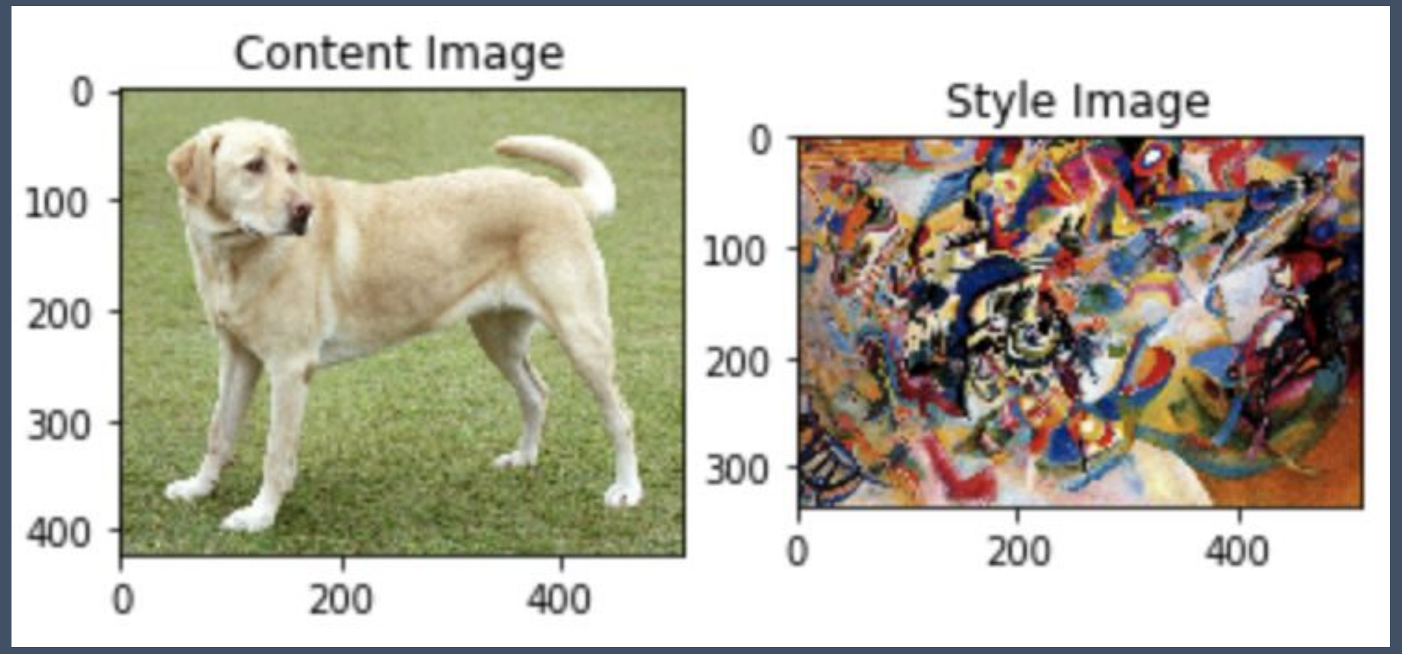
Idea: use outputs of one or more layers of a network trained on a different task as generic feature detectors. Train a new shallow model on these features.

Assumes that $D_S = D_T$



Colab Demo

Style Transfer



```
import tensorflow_hub as hub
hub_handle = 'https://tfhub.dev/google/magenta/arbitrary-image-stylization-v1-256/1'
hub_module = hub.load(hub_handle)
stylized_image = hub_module(tf.constant(content_image), tf.constant(style_image))[0]
tensor_to_image(stylized_image)
```

tensorflow.org/tutorials/generative/style_transfer

Text Classification

```
import tensorflow as tf
import tensorflow_hub as hub
embedding = "https://tfhub.dev/google/tf2-preview/gnews-swivel-20dim/1"
hub_layer = hub.KerasLayer(embedding, input_shape=[], dtype=tf.string, trainable=True)
```

https://www.tensorflow.org/tutorials/keras/text_classification_with_hub

Text Classification

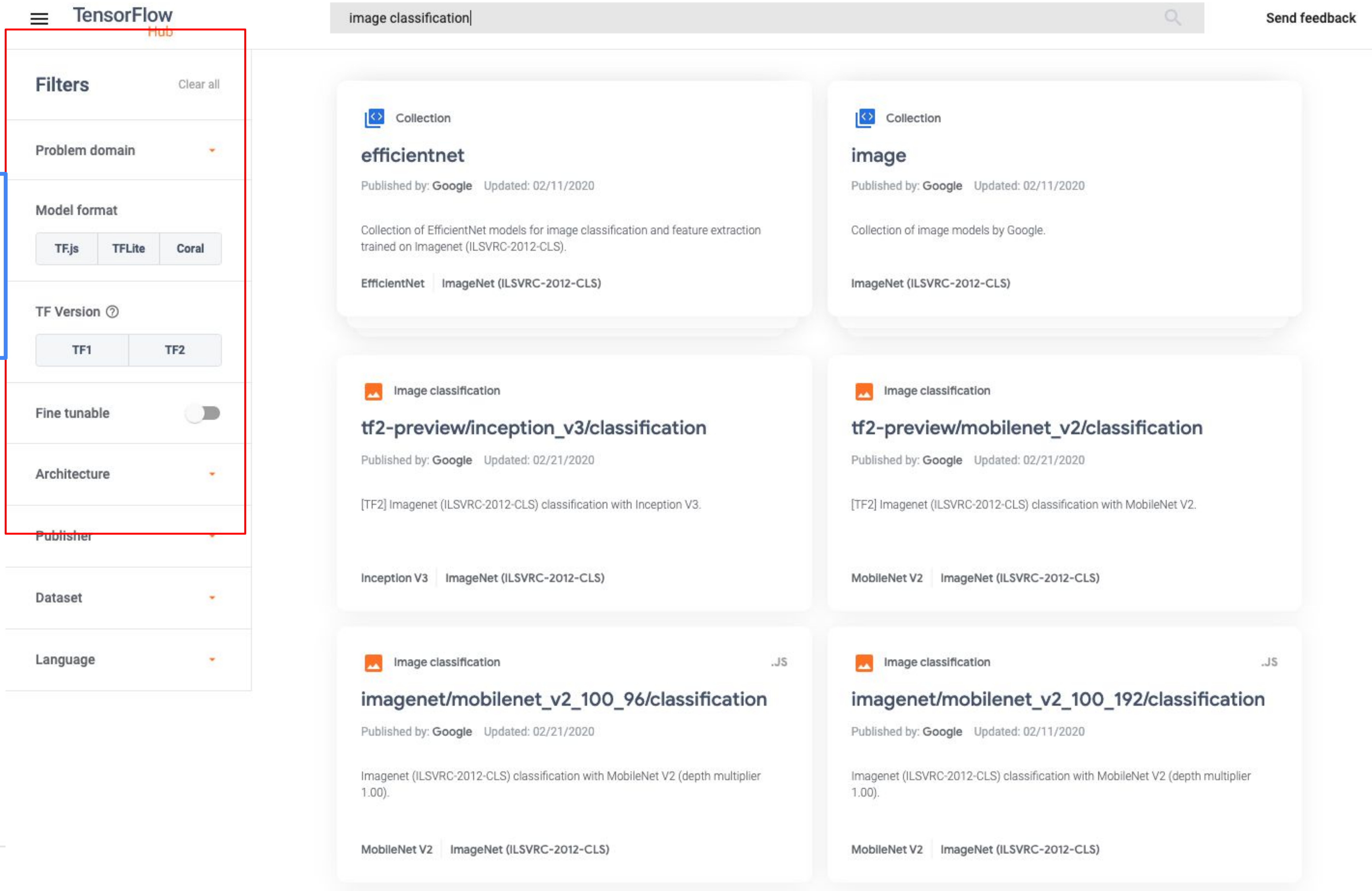
```
import tensorflow as tf
import tensorflow_hub as hub
embedding = "https://tfhub.dev/google/tf2-preview/gnews-swivel-20dim/1"
hub_layer = hub.KerasLayer(embedding, input_shape=[], dtype=tf.string, trainable=True)

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

model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])
history = model.fit(train_data.shuffle(10000).batch(512), epochs=20,
                    validation_data=validation_data.batch(512), verbose=1)
```

https://www.tensorflow.org/tutorials/keras/text_classification_with_hub

What’s New

Better Searching for Directories

Improved searching methods for a better experience



Expanded Support for TF formats

Tensorflow Lite + Metadata

Additional metadata added to TF-Lite models

Makes deployment on embedded devices easier

TensorFlow.js

New models for face and hand tracking (published by MediaPipe)

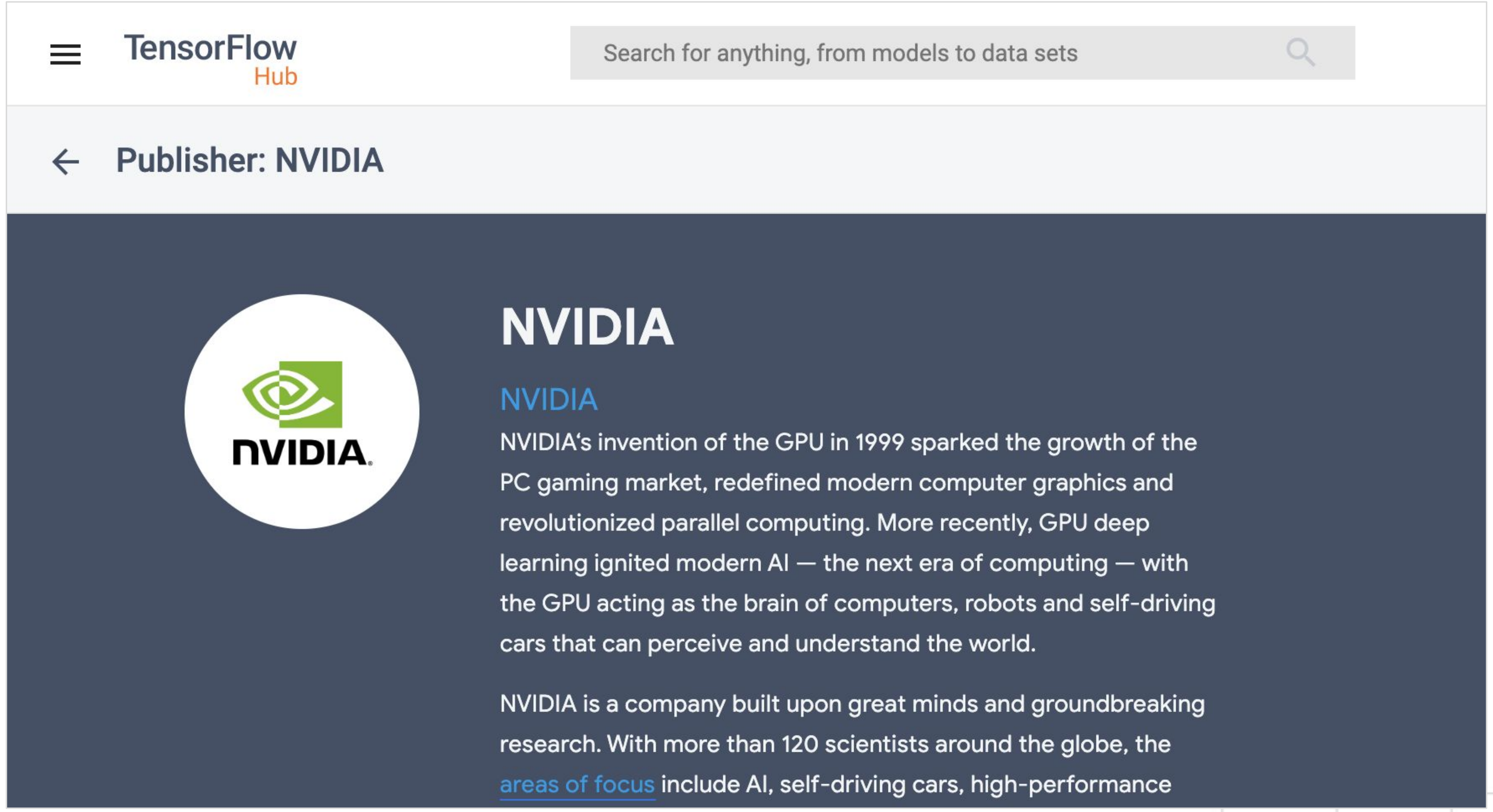
Adding more text models for interactive web apps

With the Help of the Community...

More features will be made compatible with TF after being trained, built and deployed by the community

Some Examples are:

- DeepMind
- Google
- Microsoft AI for Earth
- NVIDIA
- The Metropolitan Museum of Art
- Global Biodiversity Information Facility
- Kaggle
- And more...



Problem Domains

The TensorFlow Hub lets you search and discover hundreds of trained, ready-to-deploy machine learning models in one place. This is made easier by the option to browse by problem domains.

TensorFlow Hub is a repository for machine learning models.

From image classification, text embeddings, audio, and video action recognition, TensorFlow Hub is a space where you can browse trained models and datasets from across the TensorFlow ecosystem. Use it to:

1. Find trained models for transfer learning to save time on training
2. Publish your own models
3. Deploy models on device and in the browser

Search by Problem Domains

On Tensorflow hub you can discover models and collections related to these different problem domains.

Image

Video

Text

Text

Visit https://tfhub.dev/

Text

- Classification (2)
- Embedding (146)
- Generation (8)
- Language … (44)
- Question ans… (3)

Image

- Augmentation (6)
- Classificati… (189)
- Classification… (1)
- Depth estima… (1)
- Classifier (5)
- Generator (30)
- Object dete… (55)
- Others (1)
- Pose detect… (12)
- RNN agent (10)

Video

- Classification (6)
- Generation (5)
- Text (2)

Audio

- Embedding (4)
- Event classifi… (1)
- Pitch extraction (1)

Sample Models

Text Domain - Bidirectional Encoder Representations from Transformers (BERT)
https://www.tensorflow.org/hub/tutorials/bert_experts?hl=en

```
BERT_MODEL = "https://tfhub.dev/google/experts/bert/wiki_books/1" # @param {type: "string"}
["https://tfhub.dev/google/experts/bert/wiki_books/1",
"https://tfhub.dev/google/experts/bert/wiki_books/mnli/1",
"https://tfhub.dev/google/experts/bert/wiki_books/qnli/1",
"https://tfhub.dev/google/experts/bert/wiki_books/qqp/1",
"https://tfhub.dev/google/experts/bert/wiki_books/squad2/1",
"https://tfhub.dev/google/experts/bert/wiki_books/sst2/1",
"https://tfhub.dev/google/experts/bert/pubmed/1", "https://tfhub.dev/google/experts/bert/pubmed/squad2/1"]

MAX_SEQUENCE_LENGTH =  512

bert = hub.load(BERT_MODEL)
vocab_path = bert.vocab_file.asset_path.numpy()
tokenizer = tokenization.FullTokenizer(vocab_path, do_lower_case=bert.do_lower_case)
inputs = build_inputs(tokenizer, sentences, MAX_SEQUENCE_LENGTH)
pooled_output, sequence_output = bert(inputs)
```

Sample Models

Image Domain - Object Detection
https://www.tensorflow.org/hub/tutorials/tf2_object_detection?hl=en


kite: 57.0%
kite: 79.0%
kite: 70.0%
kite: 65.0%
kite: 47.0%
kite: 67.0%
kite: 70.0%
kite: 41.0%
kite: 33.0%
person: 57.0%
person: 53.0%
person: 59.0%
person: 43.0%
person: 50.0%
person: 66.0%
surfboard: 35.0%
person: 71.0%
person: 76.0%
0
100
200
300
400
500
600
700
800

Sample Models

Image Domain - Image Classifier (MobileNet V2)
https://www.tensorflow.org/hub/tutorials/tf2_image_retraining?hl=en

```
module_selection = ("mobilenet_v2_100_224", 224)
handle_base, pixels = module_selection
MODULE_HANDLE ="https://tfhub.dev/google/imagenet/{}/feature_vector/4".format(handle_base)
IMAGE_SIZE = (pixels, pixels)
print("Using {} with input size {}".format(MODULE_HANDLE, IMAGE_SIZE))

BATCH_SIZE = 32
```

```
True label: daisy
Predicted label: daisy
```

Thanks you

Slide:

https://bit.ly/2GzrrjL