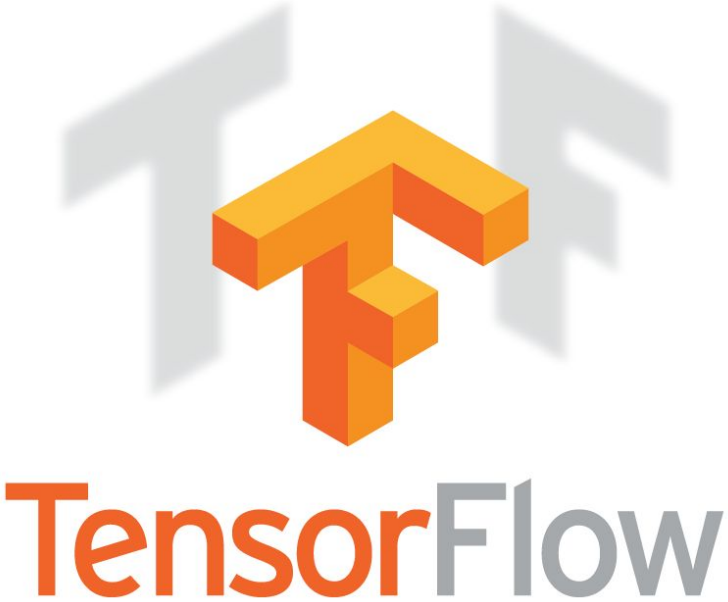
An Introduction to (Modern) TensorFlow

Simran Khanuja, Ameya Daigavane

Google Research

A multidimensional array.

A graph of operations.



- Machine learning library; especially popular for **deep learning.**
- Extensively used for research **and** production.
- **Open-source**: Apache 2.0 license.

Contents

- Introductory Colab for TensorFlow Basics
- Two applications with TensorFlow 2.0:
  - **Natural Language Processing**: BERT Fine-tuning on a sentiment review dataset.
  - **Graph Neural Networks**: Sorting linked-lists.
- Questions and Discussion!

Introductory Colab for TensorFlow Basics

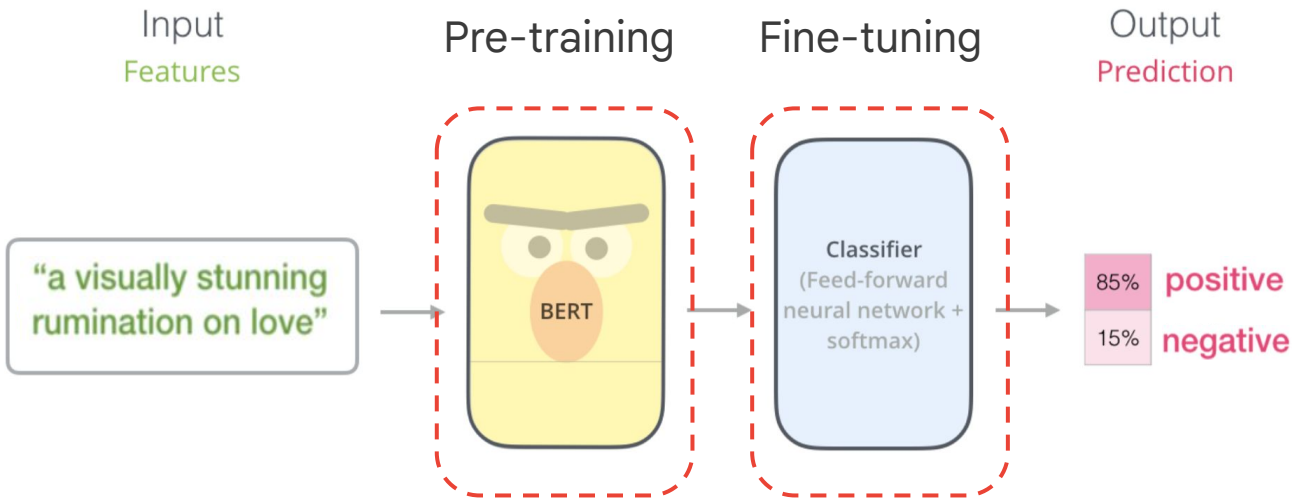
https://colab.research.google.com/drive/1zD3I0iO6ont2fiGS0YPfVAoMdGP1-KHr?usp=sharing

- What are Tensors?
- What operations can I do with them?
- What is auto-differentiation?

Natural Language Processing In TensorFlow

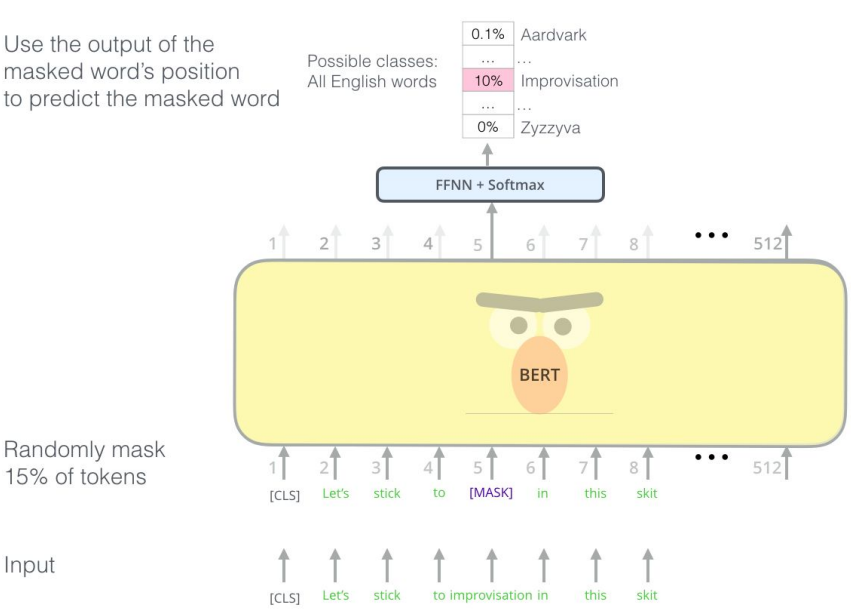
Today’s Application: Building a Sentiment Classifier using BERT!

What is BERT?

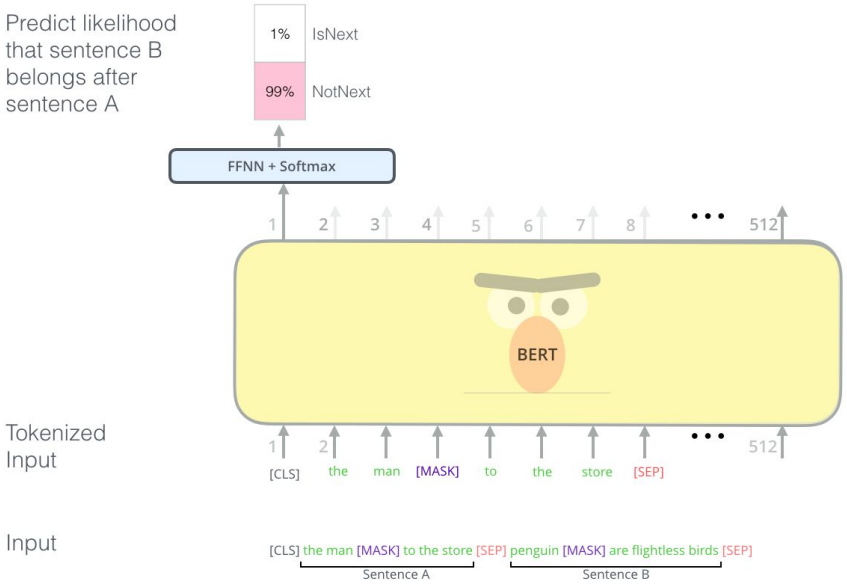


Source

BERT Pre-training



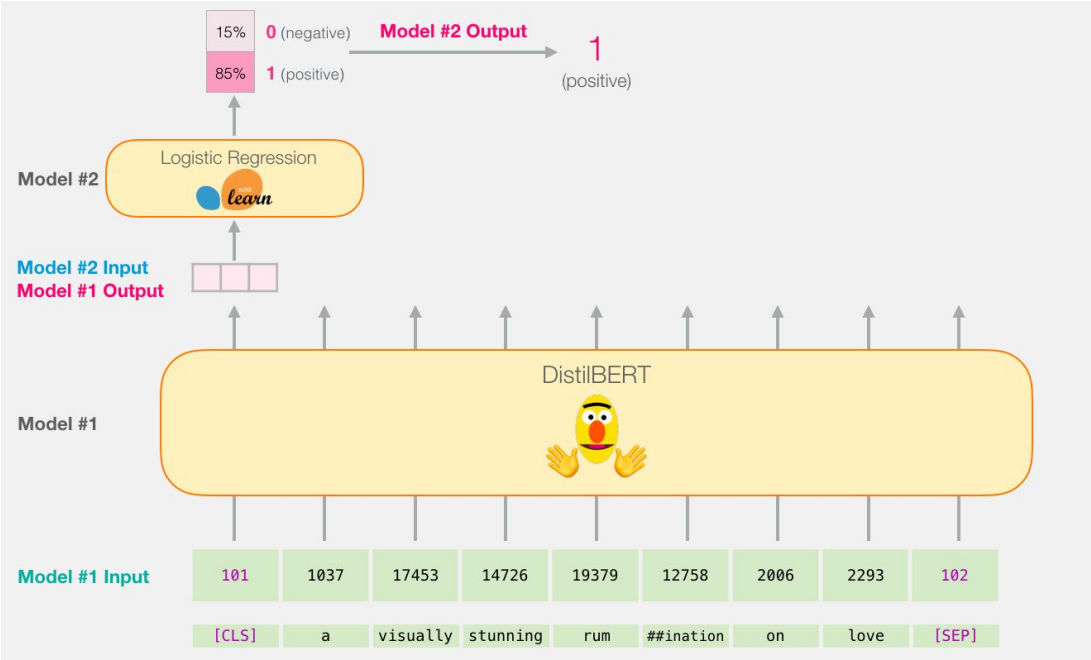
Masked Language Modeling



Next Sentence Prediction

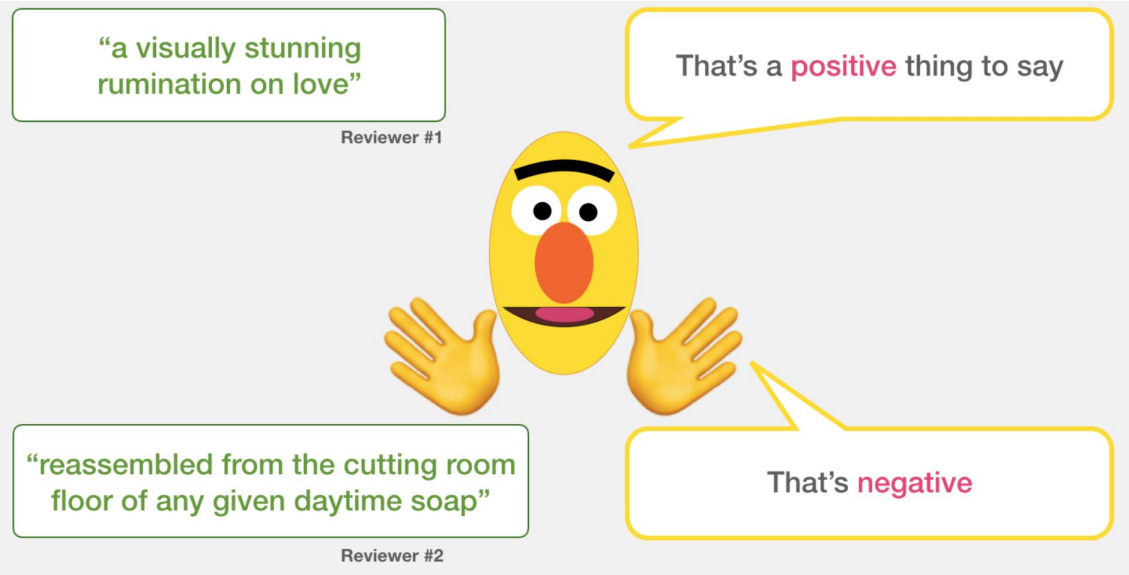
Source

BERT Fine-tuning



Source

Building a Sentiment Classifier!



https://colab.research.google.com/drive/1121vEUkIHh3_iwIbnr8nGnIaq0lDe6D3?usp=sharing

Source

Graph Neural Networks In TensorFlow

What are Graph Neural Networks?

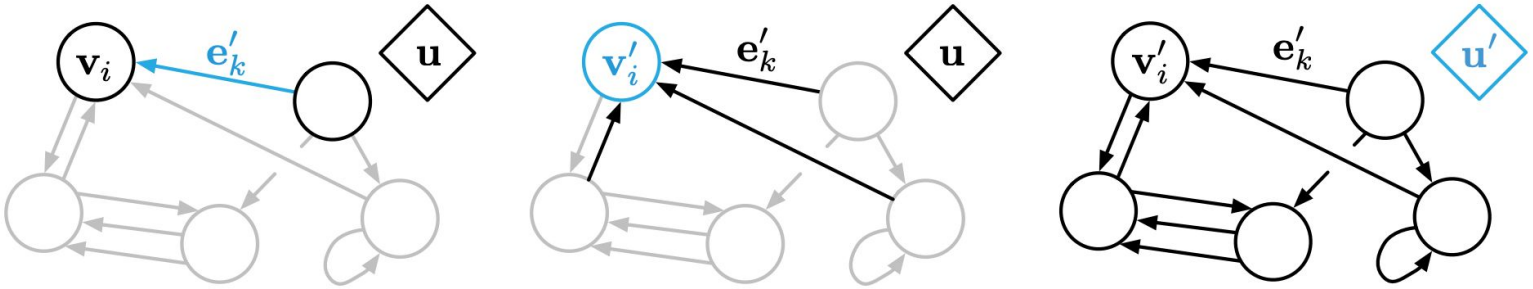
- Family of neural networks that can operate on graph-structured data:
  - Social networks
  - Protein-protein interactions
  - Physical simulations
  - Traffic networks
  - ... many many more. Anywhere you have a graph, you can use a GNN!
- A primer on understanding Graph Neural Networks:
  - https://drafts.distill.pub/distillpub/post--understanding-gnns/

Graph Neural Networks

- Most graph neural networks are built of the same building blocks:
  - Transform
    - Apply some (common) function to current node features.
    - This function is generally a neural network.
  - Aggregate
    - Aggregate neighbouring features into every node.
    - Every node is now updated.
- Similar for graphs with edges and global attributes.

Graph Networks

Today, we will be looking at Graph Networks, one (broad) family of GNNs.



(a) Edge update (b) Node update (c) Global update

https://arxiv.org/pdf/1806.01261.pdf

Graph Networks in TensorFlow

GraphNets Library from DeepMind: https://github.com/deepmind/graph_nets

```
import graph_nets as gn
import sonnet as snt

# Provide your own functions to generate graph-structured data.
input_graphs = get_graphs()

# Create the graph network.
graph_net_module = gn.modules.GraphNetwork(
    edge_model_fn=lambda: snt.nets.MLP([32, 32]),
    node_model_fn=lambda: snt.nets.MLP([32, 32]),
    global_model_fn=lambda: snt.nets.MLP([32, 32]))

# Pass the input graphs to the graph network, and return the output graphs.
output_graphs = graph_net_module(input_graphs)
```

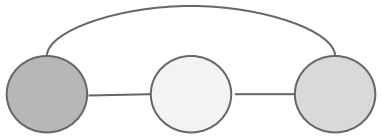
Sorting Linked-Lists with GNNs: First Approach

- The input is a linked-list of numbers.
  - Every number is a node.
  - Every pointer is an edge.
- The task is to sort this linked list in ascending order.

We can think of this as a graph-to-graph problem, keeping the connectivity of the graph fixed.

Sorting Linked-Lists with GNNs: Second Approach

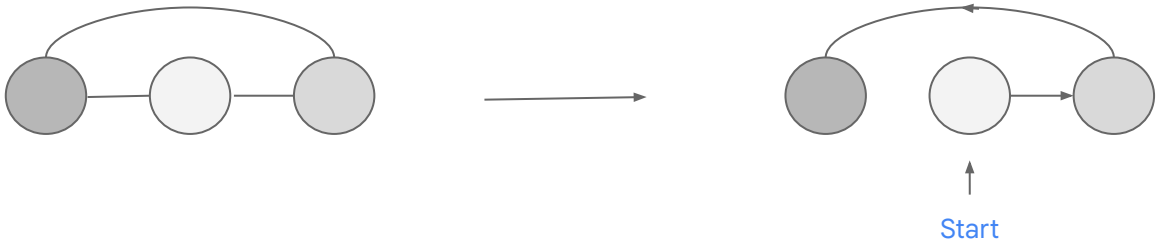
- A problem with our first approach?
  - Information propagation only occurs along the linked-list.
  - Remember: long-term dependencies in RNNs!
- Instead, connect all nodes together to create a fully-connected graph.
  - Easier information flow across nodes.



Transformed linked list

Sorting Linked-Lists with GNNs: Second Approach

- Our network will now predict two things:
  - The starting node
  - At each node, the edge to the next node in the sorted order.



Sorting Linked-Lists with GNNs: Colab

https://colab.research.google.com/drive/1Z7zPKoD7MJBh_vf1hzdHPrquW8ncuREP?usp=sharing

- Construct (synthetic) data pipeline.
- Construct GraphNets model.
- Define loss functions and training loop (with tf.GradientTape).
- Optimize loss.
- Evaluate!