Tutorial 6 - Deep Learning

Victoria Ajila, MASc Computer Engineering

Carleton University

Monday 1st November, 2021

Asynchronous Tutorial

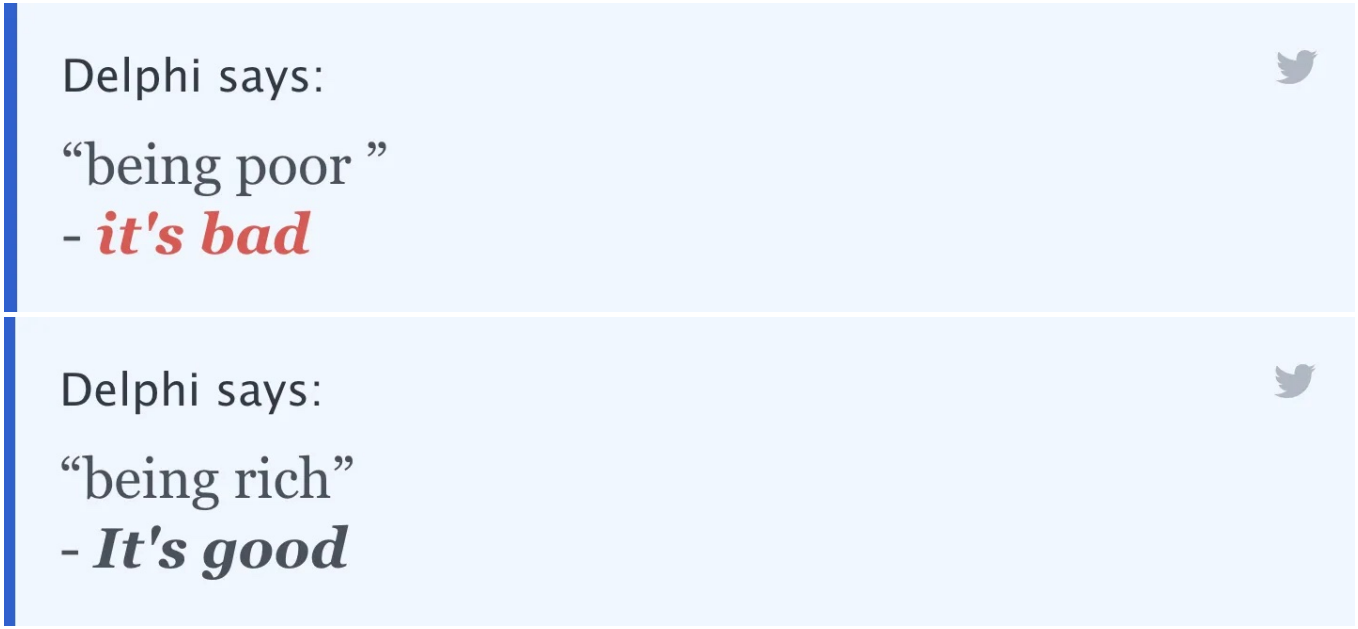
To watch and following along at your leisure

ML Weekly

Recent news events from the ML community

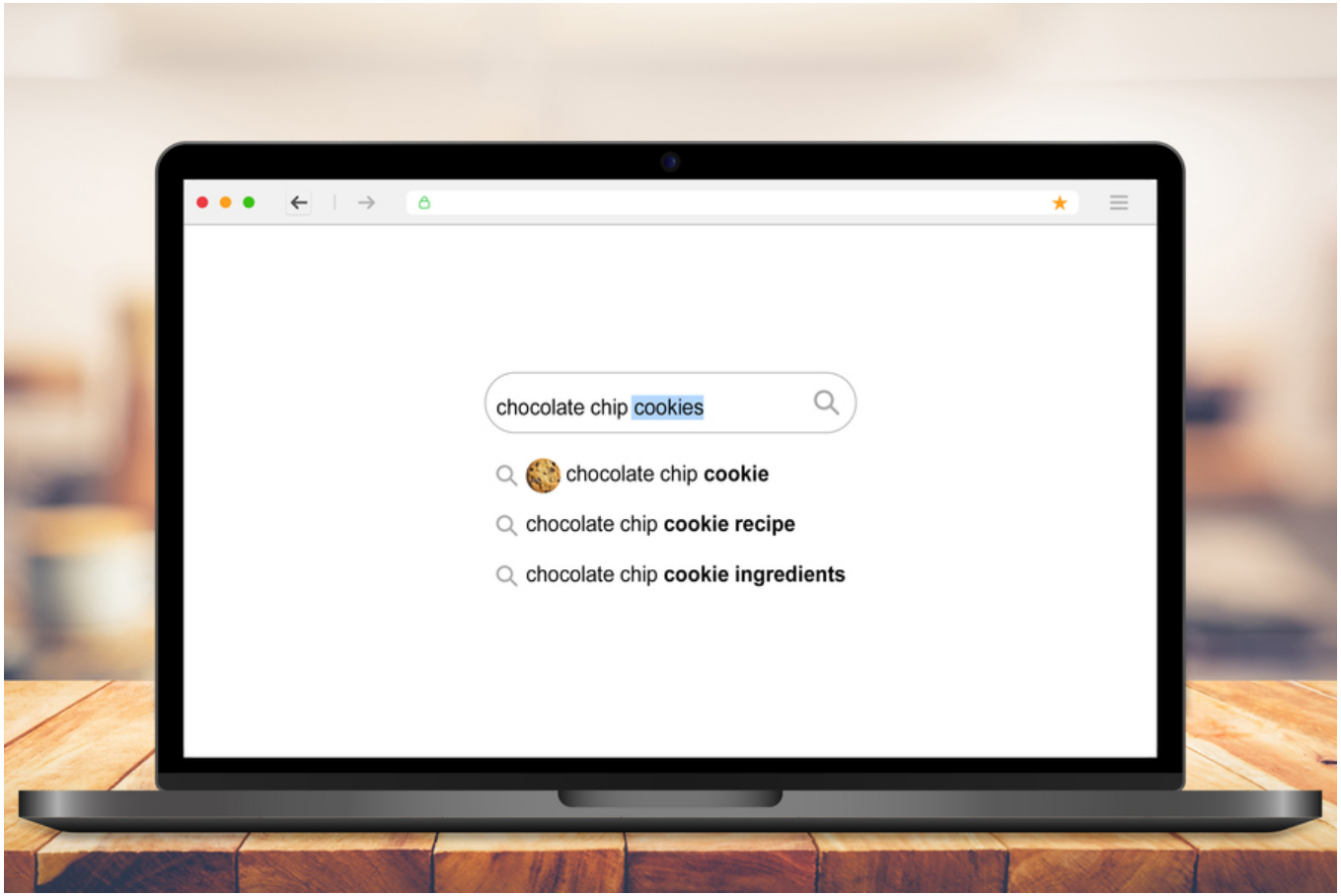
ML Weekly

1. **(AI)** The AI oracle of Delphi uses the problems of Reddit to offer dubious moral advice



ML Weekly

1. **(AI)** The AI oracle of Delphi uses the problems of Reddit to offer dubious moral advice
2. **(NLP)** Artificial intelligence sheds light on how the brain processes language

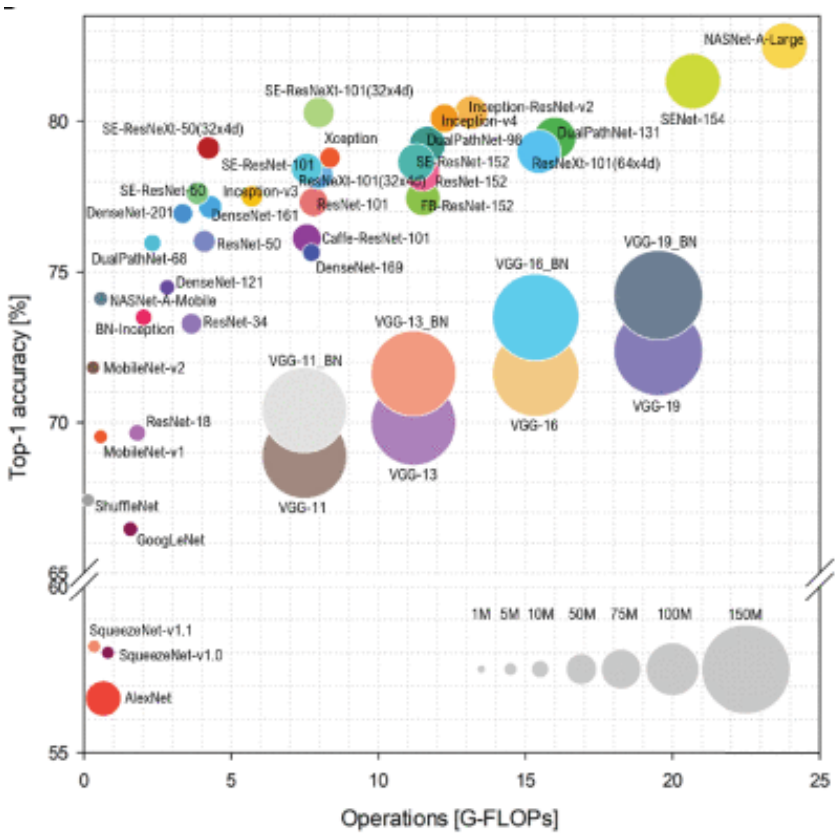


ML Weekly

1. **(AI)** The AI oracle of Delphi uses the problems of Reddit to offer dubious moral advice
2. **(NLP)** Artificial intelligence sheds light on how the brain processes language
3. **(ML)** A radical new technique lets AI learn with practically no data

ML Weekly

1. **(AI)** The AI oracle of Delphi uses the problems of Reddit to offer dubious moral advice
2. **(NLP)** Artificial intelligence sheds light on how the brain processes language
3. **(ML)** A radical new technique lets AI learn with practically no data
4. **(Vision)** Benchmark analysis of representative deep neural network architectures



Tutorial Intuition

Building an Intuition for the Concepts of this Tutorial

Introducing the Keras API: Deep Network in < 30 Lines

The core data structures of Keras are **layers** and **models**. The simplest type of model is the **Sequential model**, a linear stack of layers.

Algorithm 1 Pseudocode for Building a Deep Learning (Vision) Model

Input: model parameters
Output: model object implementing architecture

```
initialized model object, m
m ← add input layer
for each hidden layer, h_i, in {h_1, h_2, ..., h_n} do
    m ← add a convolutional layer
    m ← add an activation layer (optional)
    m ← add a normalization layer (optional)
    m ← add a pooling layer (optional)
    m ← add a dropout layer (optional)
end for
m ← add output layer
return m
```

Introducing the Keras API

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense

model = Sequential() # a linear stack of layers

# stacking layers is as easy as .add():
#
#

# configure model learning process with .compile():
#
#
#

# (Optional) further configure your optimizer.
#
#
#

# x_train and y_train are Numpy arrays akin to Scikit-Learn API.
#

# Evaluate your test loss and metrics in one line:
#

# Or generate predictions on new data:
#
```

Introducing the Keras API

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense

model = Sequential() # a linear stack of layers

# stacking layers is as easy as .add():
model.add(Dense(units=64, activation='relu'))
model.add(Dense(units=10, activation='softmax'))

# configure model learning process with .compile():
#
#
#

# (Optional) further configure your optimizer.
#
#
#

# x_train and y_train are Numpy arrays akin to Scikit-Learn API.
#

# Evaluate your test loss and metrics in one line:
#

# Or generate predictions on new data:
#
```

Introducing the Keras API

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense

model = Sequential() # a linear stack of layers

# stacking layers is as easy as .add():
model.add(Dense(units=64, activation='relu'))
model.add(Dense(units=10, activation='softmax'))

# configure model learning process with .compile():
model.compile(loss='categorical_crossentropy',
              optimizer='sgd',
              metrics=['accuracy'])

# (Optional) further configure your optimizer.
model.compile(loss=keras.losses.categorical_crossentropy,
              optimizer=keras.optimizers.SGD(learning_rate=0.01,
              momentum=0.9, nesterov=True))

# x_train and y_train are Numpy arrays akin to Scikit-Learn API.
#

# Evaluate your test loss and metrics in one line:
#

# Or generate predictions on new data:
#
```

Introducing the Keras API

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense

model = Sequential() # a linear stack of layers

# stacking layers is as easy as .add():
model.add(Dense(units=64, activation='relu'))
model.add(Dense(units=10, activation='softmax'))

# configure model learning process with .compile():
model.compile(loss='categorical_crossentropy',
              optimizer='sgd',
              metrics=['accuracy'])

# (Optional) further configure your optimizer.
model.compile(loss=keras.losses.categorical_crossentropy,
              optimizer=keras.optimizers.SGD(learning_rate=0.01,
              momentum=0.9, nesterov=True))

# x_train and y_train are Numpy arrays akin to Scikit-Learn API.
model.fit(x_train, y_train, epochs=5, batch_size=32)

# Evaluate your test loss and metrics in one line:
#

# Or generate predictions on new data:
#
```

Introducing the Keras API

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense

model = Sequential() # a linear stack of layers

# stacking layers is as easy as .add():
model.add(Dense(units=64, activation='relu'))
model.add(Dense(units=10, activation='softmax'))

# configure model learning process with .compile():
model.compile(loss='categorical_crossentropy',
              optimizer='sgd',
              metrics=['accuracy'])

# (Optional) further configure your optimizer.
model.compile(loss=keras.losses.categorical_crossentropy,
              optimizer=keras.optimizers.SGD(learning_rate=0.01,
              momentum=0.9, nesterov=True))

# x_train and y_train are Numpy arrays akin to Scikit-Learn API.
model.fit(x_train, y_train, epochs=5, batch_size=32)

# Evaluate your test loss and metrics in one line:
loss_and_metrics = model.evaluate(x_test, y_test, batch_size=128)

# Or generate predictions on new data:
classes = model.predict(x_test, batch_size=128)
```

Into the Notebooks we Go...

We will cover one new notebook today!

1. Tutorial 6 - Deep Learning

Tutorial 6 - Deep Learning

Victoria Ajila, MASc Computer Engineering

Carleton University

Monday 1st November, 2021