Google Cloud

Text Classification

Advanced ML with TensorFlow on GCP

End-to-End Lab on Structured Data ML

Production ML Systems

Image Classification Models

Sequence Models

Recommendation Systems

Agenda

Working with text

Text classification

Python versus native TensorFlow

Working with text

Natural language is a sequence

Fence the over jumped fox brown quick the.

The quick brown fox jumped over the fence.

Agenda

Working with text

Text classification

Python versus native TensorFlow

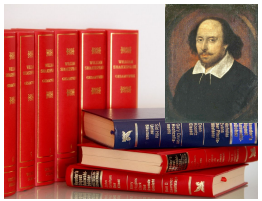
Text classification

1 Should we flag this email as spam?

2 Is the customer satisfied with our product?

3 Is this document related to a specific project?

4 Was this play written by Shakespeare?



Problem Statement: Given the title of an article, figure out the publication that the article appeared in

"Supreme Court to hear major case on Partisan Districts"

Text classification model

Predict publisher

- **New York Times**
- TechCrunch
- GitHub

Working with text data

Supreme	Court	to	hear	major	case	…
0.3	0.6	0.1	1.9	0.3	0.6	0.1
1.2	2.1	1.9	1.5	1.2	2.1	1.9
1.6	2.2	2.1	0.3	1.6	2.2	2.1
0.9	1.3	1.8	2.2	0.9	1.3	1.8
0.3	1.5	0.8	1.8	0.3	1.5	0.8
0.5	1.0	0.9	1.1	0.5	1.0	0.9
1.6	1.4	3.2	0.4	1.6	1.4	3.2
2.3	0.6	1.1	1.6	2.3	0.6	1.1

Text classification model

Predict publisher

- **New York Times**
- TechCrunch
- GitHub

Convert sentences into a numeric representation

1. Create a mapping from each word to a unique integer.
2. Encode each sentence as a sequence of these integers.
3. Pad each sequence to a constant length.
4. Convert each integer into an embedded representation with meaningful magnitude.

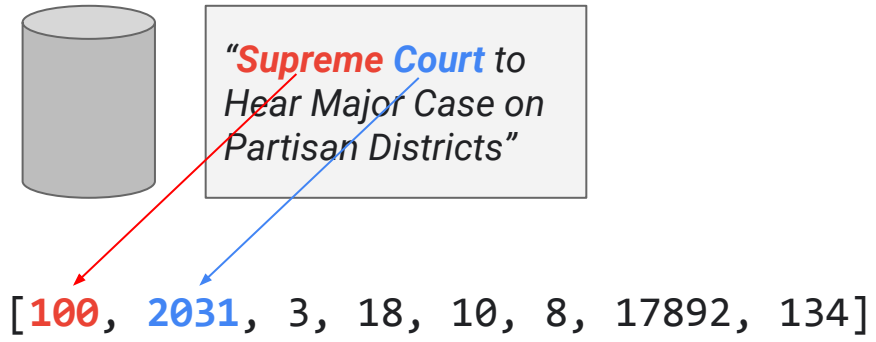
(1) Create a mapping from each word to a unique integer


```
{Supreme: 100,
 Court: 2301,
 The: 2,
 Major: 10
 ....}
```

```
from tensorflow.python.keras.preprocessing import text

tokenizer = text.Tokenizer(num_words=TOP_K) # only encode TOP_K most frequent words
tokenizer.fit_on_texts(titles) # titles is a python list of strings

# Save mapping to use during prediction time
pickle.dump(tokenizer, open('tokenizer.pickled', 'wb'))
```

(2) Encode each sentence as a sequence of these integers

```
x = tokenizer.texts_to_sequences(texts)
```



(3) Pad each sequence to a constant length

```
from tensorflow.python.keras.preprocessing import sequence

x = sequence.pad_sequences(x, maxlen=MAX_SEQUENCE_LENGTH)
```

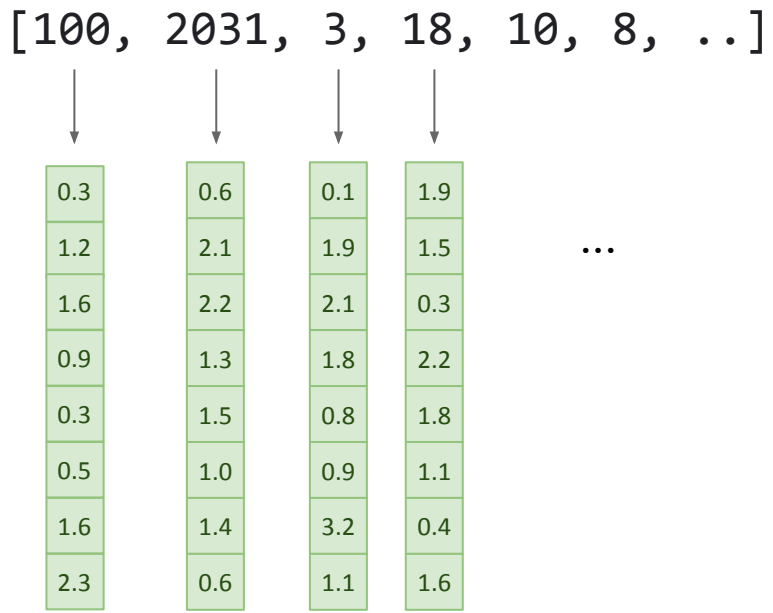
MAX_SEQUENCE_LENGTH = 15

```
[100, 2031, 3, 18, 10, 8, 17892, 134]
```

Padded to 15

```
[100, 2031, 3, 18, 10, 8, 17892, 134, 0, 0, 0, 0, 0, 0, 0]
```

(4) Convert each integer into an embedded representation with meaningful magnitude



(4) Convert each integer into an embedded representation with meaningful magnitude (Keras)

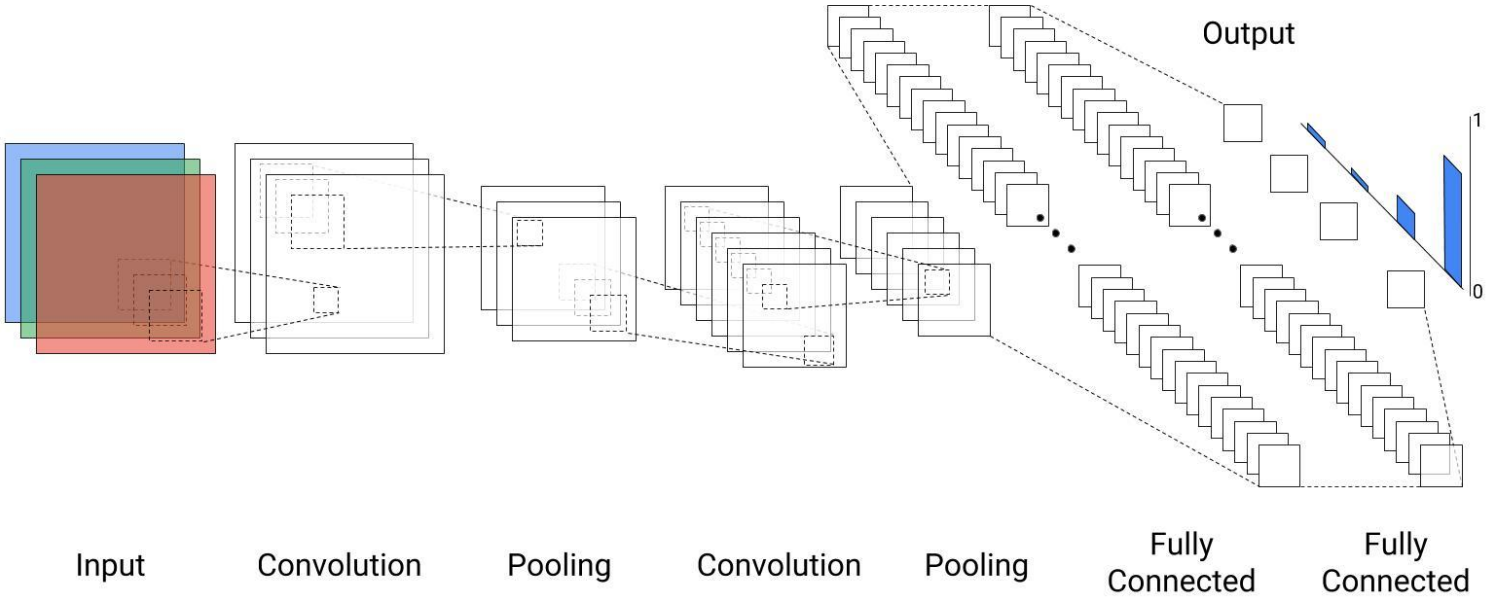
```
from tensorflow.python.keras import models
from tensorflow.python.keras.layers import Embedding

model = models.Sequential()
model.add(Embedding(input_dim=vocabulary_size,
                    output_dim=embedding_dim,
                    input_length=MAX_SEQUENCE_LENGTH))
```

(4) Convert each integer into an embedded representation with meaningful magnitude (TensorFlow)

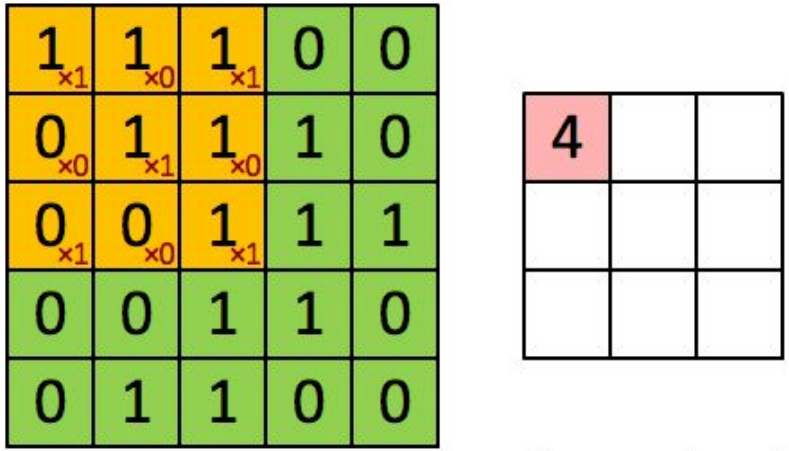

```
one_hot_word = tf.feature_column.categorical_column_with_vocabulary_list(
                    'word', vocabulary_list=englishWords)

embedded_word = tf.feature_column.embedding_column(one_hot_word, embedding_dim)
```

What are our choices in building the model?

CNNs have some good properties for text classification



CNNs have some good properties for text classification



Image

Convolved feature

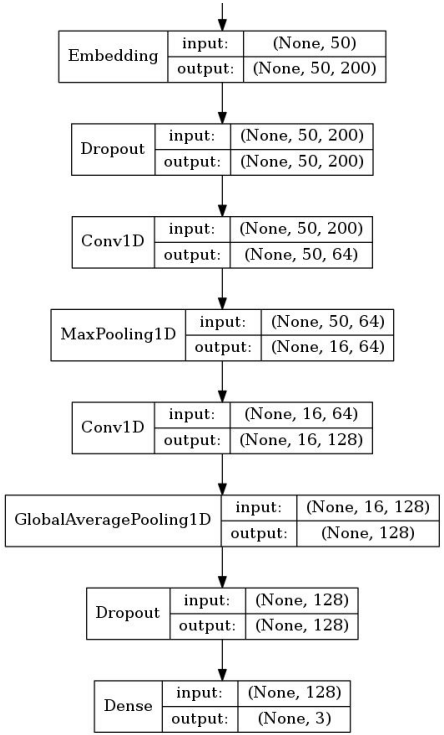
CNNs have some good properties for text classification

“Supreme Court to hear major case on Partisan Districts”

Supreme	Court	to	hear	major	case	…
0.3	0.6	0.1	1.9	0.3	0.6	0.1
1.2	2.1	1.9	1.5	1.2	2.1	1.9
1.6	2.2	2.1	0.3	1.6	2.2	2.1
0.9	1.3	1.8	2.2	0.9	1.3	1.8
0.3	1.5	0.8	1.8	0.3	1.5	0.8
0.5	1.0	0.9	1.1	0.5	1.0	0.9
1.6	1.4	3.2	0.4	1.6	1.4	3.2
2.3	0.6	1.1	1.6	2.3	0.6	1.1

Creating CNN layers with the Keras API

```
model.add(Embedding(...))
model.add(Dropout(...))
model.add(Conv1D(...)
model.add(MaxPooling1D(...))
model.add(Conv1D(...)
model.add(GlobalAveragePooling1D())
model.add(Dropout(..))
model.add(Dense(...)
```



Converting Keras model to Estimator

```
model = models.Sequential()
…
model.compile(...)
estimator = keras.estimator.model_to_estimator(keras_model=model)
```

Lab

Text Classification

In this lab, we'll build a model to solve the text classification problem we've been talking about.

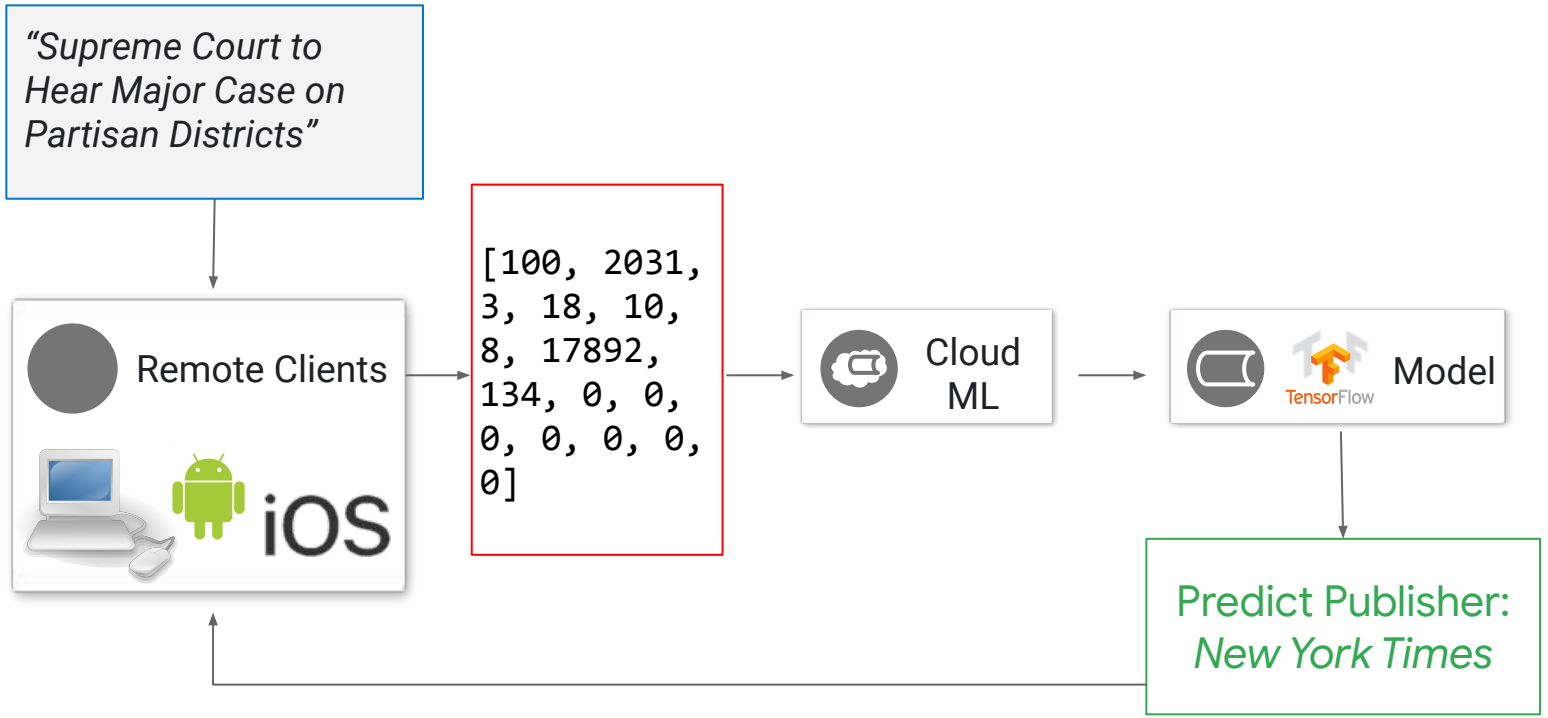
Lab Steps

1. Complete the TODOs in model.py. In particular:
   - Transform text into sequence of integers.
   - Pad sequences to a constant length.
   - Convert keras model to tf.estimator.
   - Instantiate tf.Estimator.

2. Train and Deploy model using Cloud ML Engine.

3. Feed in new article titles for prediction.

Problem: Clients forced to integerize text input



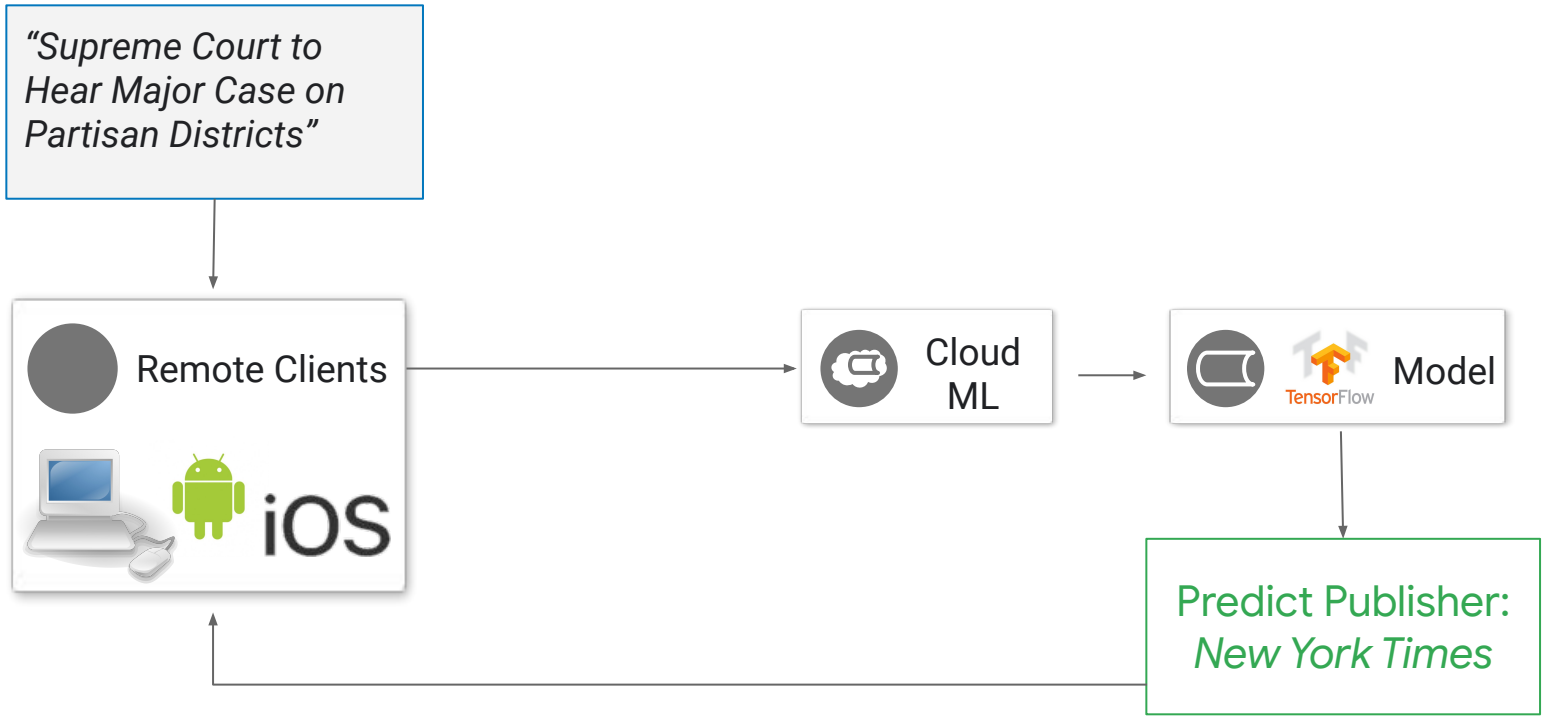
Agenda

Working with text

Text classification

Python versus native TensorFlow

Have our model API accept text directly



Why not use native TensorFlow functions at the beginning?

Can’t I just add python preprocessing code server-side?

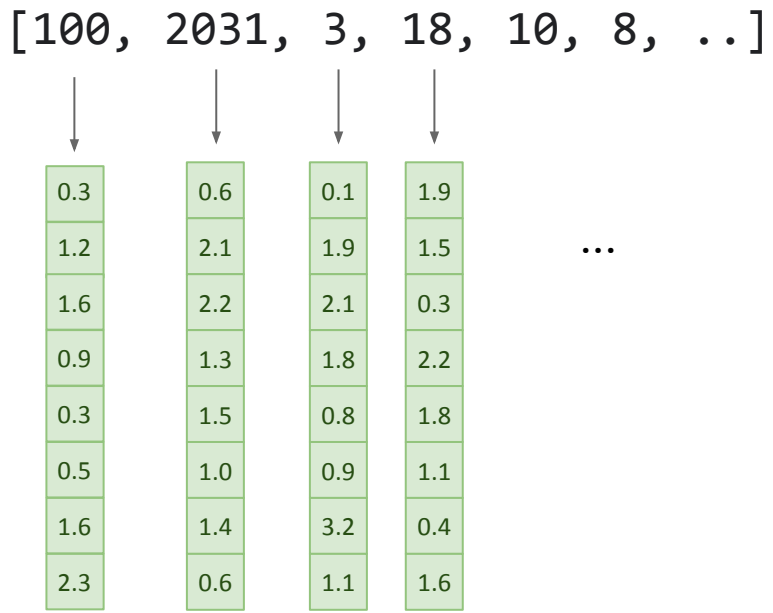
Converting from Python to TensorFlow functions

Python Function	Tensorflow Equivalent
tf.keras.preprocessing.text.Tokenizer.texts_to_sequences()	tf.contrib.lookup.index_table_from_file()
tf.keras.preprocessing.sequence.pad_sequences()	tf.pad() and tf.slice()

Demo

Text classification with native Tensorflow

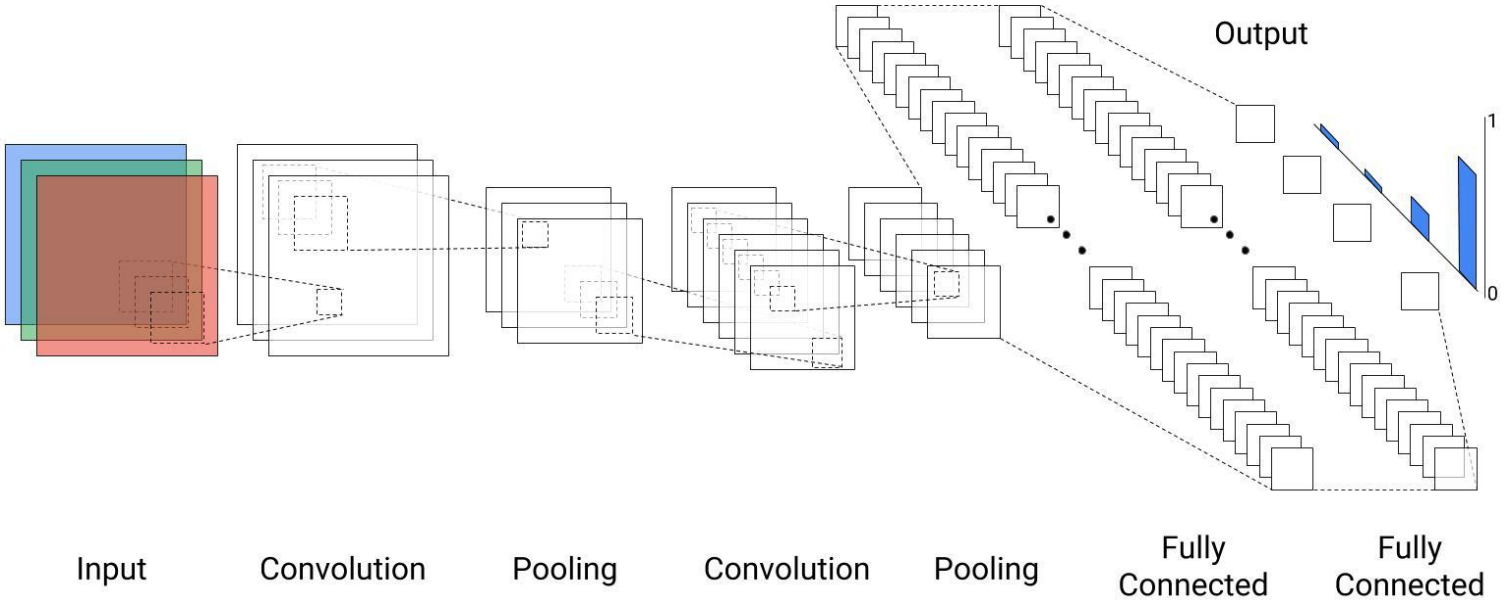
Summary: (4) Convert each integer into an embedded representation with meaningful magnitude



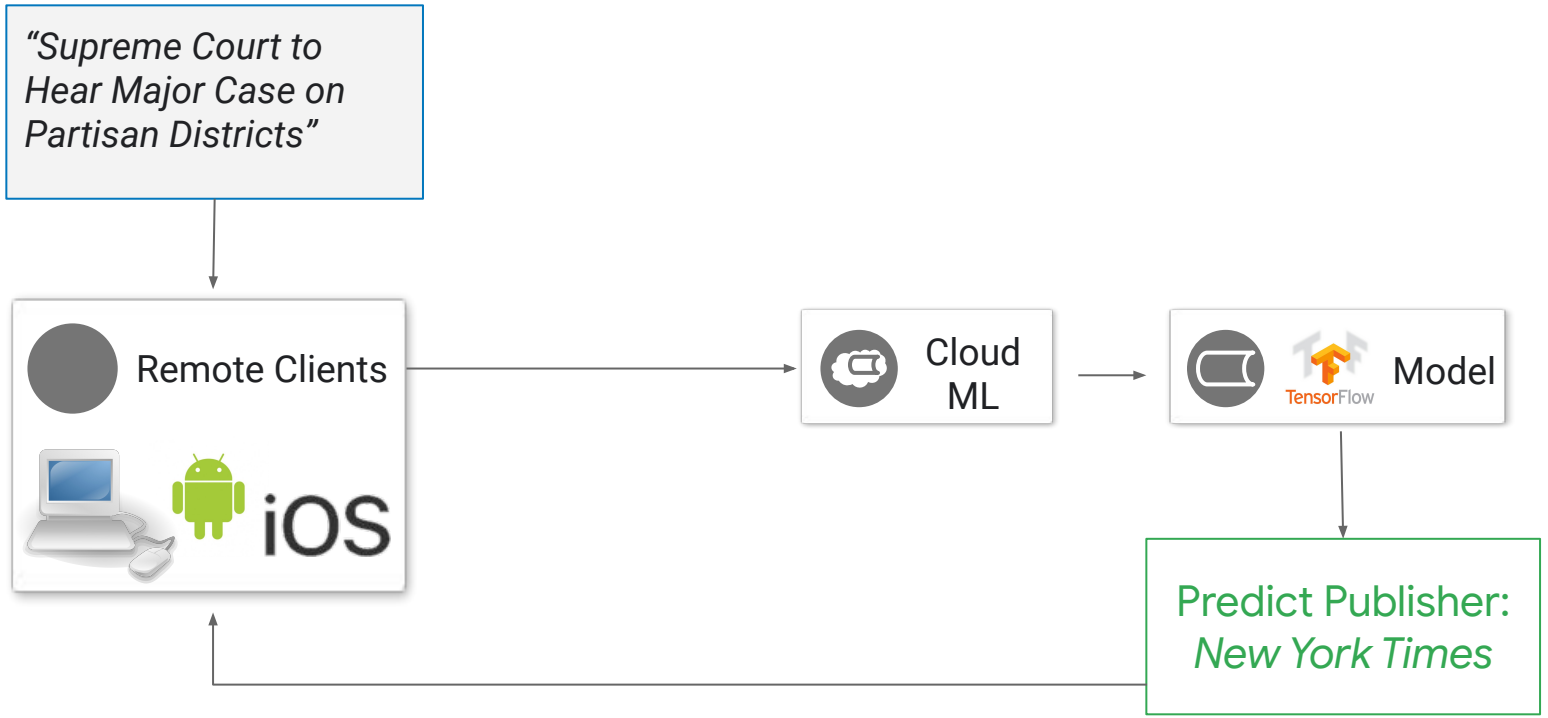
Summary: Learning an embedded representation solves both of these problems

```
from tensorflow.python.keras import models
from tensorflow.python.keras.layers import Embedding

model = models.Sequential()
model.add(Embedding(input_dim=vocabulary_size,
                    output_dim=embedding_dim,
                    input_length=MAX_SEQUENCE_LENGTH))
```

Summary: CNNs have some good properties for text classification



Summary: Have our model API accept text directly



cloud.google.com