## Google Cloud

**Encoder-Decoder Models** 



## Advanced ML with TensorFlow on GCP

End-to-End Lab on Structured Data ML

Production ML Systems

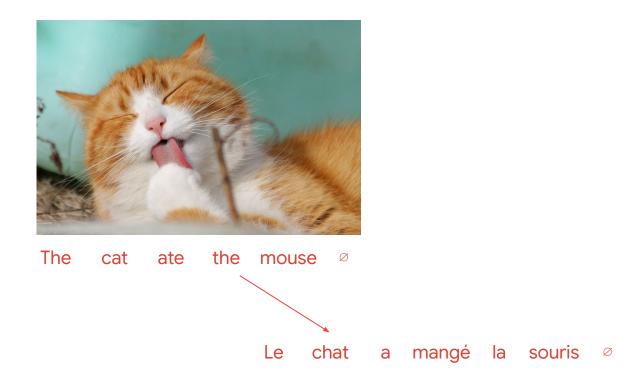
Image Classification Models

### **Sequence Models**

Recommendation Systems

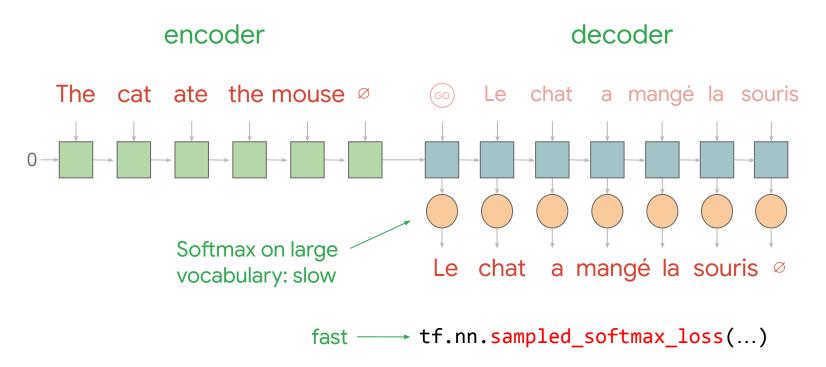


### Translate English sentence to French



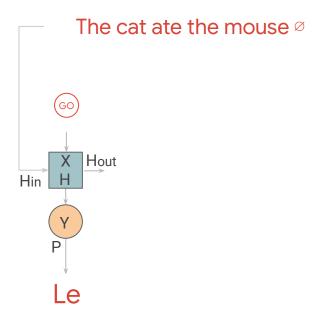


#### Encoder decoder networks





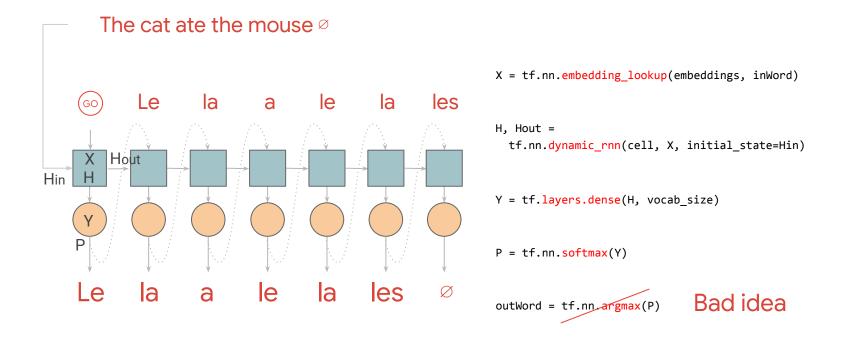
### Greedy search versus Beam search



```
X = tf.nn.embedding_lookup(embeddings, inWord)
H, Hout =
   tf.nn.dynamic_rnn(cell, X, initial_state=Hin)
Y = tf.layers.dense(H, vocab_size)
P = tf.nn.softmax(Y)
outWord = tf.nn.argmax(P)
```

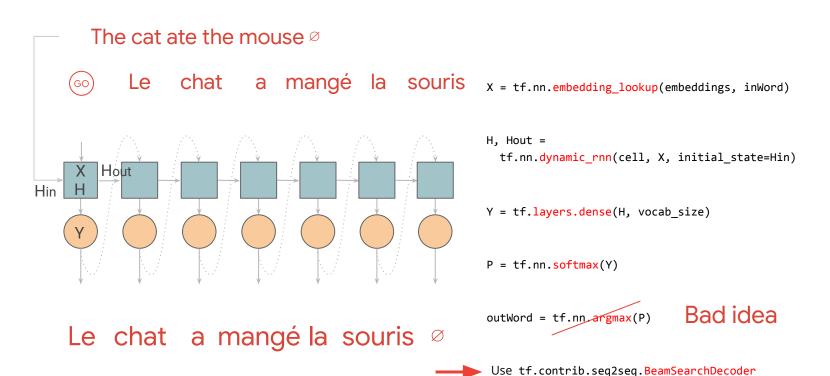


### Greedy search versus Beam search



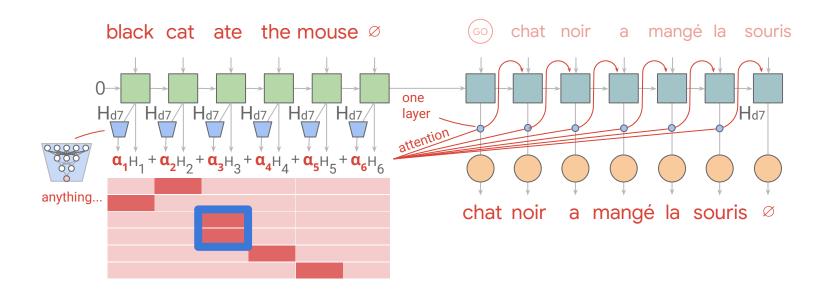


### Greedy search versus Beam search





### Improve the translation with attention network





### Four ML steps to train an encoder-decoder model

1 Embed

2 Encode

3 Attend

4 Predict

Turn words into numeric tensors.

Pretrained embeddings as a starting point, then continue to train.

Save vocabularies and embeddings for prediction.

Use multilayer GRUs to save memory and state.

Tune the GRUs carefully.

Make sure that input sequences are rich enough for the model to learn long-term rules.

Dynamically weight the outputs based on the current input word.

The number of attention heads is an important hyperparameter.

NEXT!



### TensorFlow provides APIs for many of these steps

```
Embed
x = tf.nn.embedding lookup(embeddings, sentences)
                                                                                Encode
encoder cell = tf.rnn.GRUCell(encoding dimension)
wrapped_cell = tf.rnn.DropoutWrapper(encoder_cell, output_keep_prob=p)
encoded sentences, encoder state = tf.nn.dynamic rnn(wrapped cell, x)
                                                                                 Predict
decoder_cell = tf.rnn.GRUCell(encoding_dimension)
decoder = tf.seq2seq.BeamSearchDecoder(decoder cell, embeddings,
                                      sos tokens, eos token, encoder state, beam width)
outputs, final state, = tf.seq2seq.dynamic decode(decoder, maximum iterations=max length)
```



### Translation with attention





### Replace decoder\_cell

```
Embed
x = tf.nn.embedding lookup(embeddings, sentences)
                                                                                Encode
encoder cell = tf.rnn.GRUCell(encoding dimension)
wrapped_cell = tf.rnn.DropoutWrapper(encoder_cell, output_keep_prob=p)
encoded sentences, encoder state = tf.nn.dynamic rnn(wrapped cell, x)
                                                                                 Predict
decoder = tf.seq2seq.BeamSearchDecoder(decoder_cell, embeddings,
                                         sos_tokens, eos_token, encoder_state, beam_width)
outputs, final_state, _ = tf.seq2seq.dynamic_decode(decoder, maximum_iterations=max_length)
```

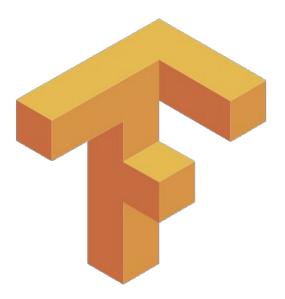


### Dropout for multi-layer GRUs

```
cells = [tf.nn.rnn cell.GRUCell(CELLSIZE) for    in range(NLAYERS)]
cells = [tf.nn.rnn cell.DropoutWrapper(cell, output keep prob=pkeep) for cell in cells]
mcell = tf.nn.rnn cell.MultiRNNCell(cells, state is tuple=False)
Hr, H = tf.nn.dynamic rnn(mcell, X, initial state=Hin)
                                                                               Dropout on
      Hin
                                                                               outputs
       X_0
                                              X_7
                                                    X^8
              X,
                    X_2
                          X_3
                                       X_6
                                                                       Typical params for character-
                                                                       based language model handling
                                                                       ~100 characters.
      GRU
             GRU
                   GRU
                          GRU
                                GRU
                                       GRU
                                             GRU
                                                    GRU
             GRU
                   GRU
                          GRU
                                GRU
                                      GRU
      GRU
                                             GRU
                                                   GRU
                                                                           CELLSIZE = 512
                                                                            NLAYERS =
      GRU
             GRU
                   GRU
                          GRU
                                GRU
                                      GRU
                                             GRU
                                                   GRU
                                                                             SEQLEN =
                                                                                          30
             H",
                    H",
                                 H"5
                                              H",
                                                             = dropout
```



### tensor2tensor comes to the rescue



github.com/tensorflow/tensor2tensor



### Use tensor2tensor to complete lines of poetry

I wandered lonely as a cloud

Model

That floats on high o'er vales and hills,



# A **Problem** ties together all the pieces that make up an ML system

```
Reuse Text2Text
@registry.register problem
class PoetryLineProblem(translate. Text2TextProblem): Problem except what
                                                         we explicitly override.
  @property
  def targeted vocab size(self):
    return 2**12 # 4096
                            Most common 4096 words form vocabulary.
  def generate samples(self, data dir, tmp dir, dataset split):
            vield {
                 "inputs": prev line,
                                         Yield inputs, targets.
                 "targets": curr line
        prev line = curr line
```



### Generate training data after the preprocessing

```
PROBLEM=poetry_line_problem
t2t-datagen \
    --t2t_usr_dir=./poetry/trainer \
    --problem=$PROBLEM \
    --data_dir=$DATA_DIR \
    --tmp_dir=$TMP_DIR
```

T2t-datagen comes with tensor2tensor and creates tf.train.Example protobufs for fast training with TensorFlow.



### Customize the sequence-to-sequence model

```
@registry.register hparams
def transformer poetry():
  hparams = transformer.transformer base()
                                                  Making the model
  hparams.num hidden layers = 2
                                                  smaller.
  hparams.hidden size = 128
  hparams.filter size = 512
                                                  Increasing dropout
  hparams.num_heads = 4
                                                  regularization.
  hparams.attention dropout = 0.6
  hparams.layer prepostprocess dropout = 0.6
                                                  Lower training rate.
  hparams.learning rate = 0.05
  return hparams
```

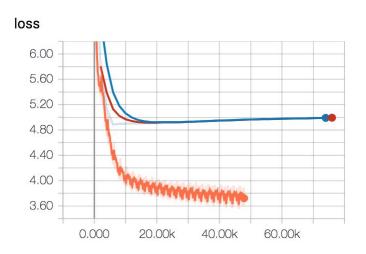


### Train the model locally and on Cloud ML Engine

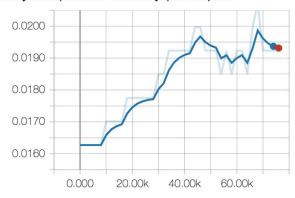
```
PROBLEM=poetry line problem
                                                  We don't have to use gcloud.
t2t-trainer \
                                                  The tensor2tensor library
  --data dir=gs://${BUCKET}/poetry/subset \
                                                  makes a REST API call to
  --t2t usr dir=./poetry/trainer \
                                                  Cloud ML Engine to submit
  --problem=$PROBLEM \
  --model=transformer \
                                                  the training job.
  --hparams set=transformer poetry \
  --output dir=$OUTDIR \
  --train steps=7500 --cloud mlengine --worker gpu=1
```



## Monitor training on TensorBoard



#### metricspoetry\_line\_problem/accuracy\_per\_sequence





# To hyperparameter tune on Cloud ML Engine, specify the hyperparameter ranges within the Problem definition



# Launch off the hyperparameter tuning job on Cloud ML Engine

```
t2t-trainer \
  --data dir=gs://${BUCKET}/poetry/subset \
  --t2t usr dir=./poetry/trainer \
  --problem=$PROBLEM \
  --model=transformer \
  --hparams set=transformer poetry \
  --output dir=$OUTDIR \
  --hparams range=transformer poetry range \
  --autotune objective='metrics-poetry line problem/accuracy per sequence' \
  --autotune maximize \
  --autotune max trials=40 \
  --autotune parallel trials=4 \
  --train steps=7500 --cloud mlengine --worker gpu=4
```



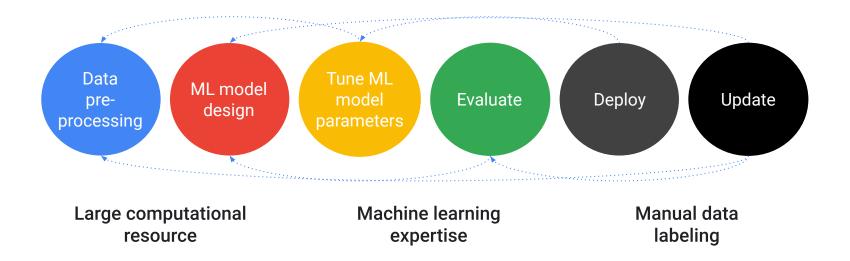
## Lab

Cloud poetry: training and hyperparameter tuning custom text models on Cloud ML Engine

## Lab Steps

- Define a Problem with tensor2tensor.
- 2. Generate training and evaluation datasets.
- 3. Customize the sequence to sequence model.
- 4. Train the model locally and on Cloud ML Engine.
- 5. Fine-tune hyperparameters.
- 6. Decode lines of poetry.
- 7. Deploy the model.

# Custom model building can be complex and time-intensive





## Use AutoML Translation<sup>BETA</sup> as a pre-built model



### ★-A AutoML Translation

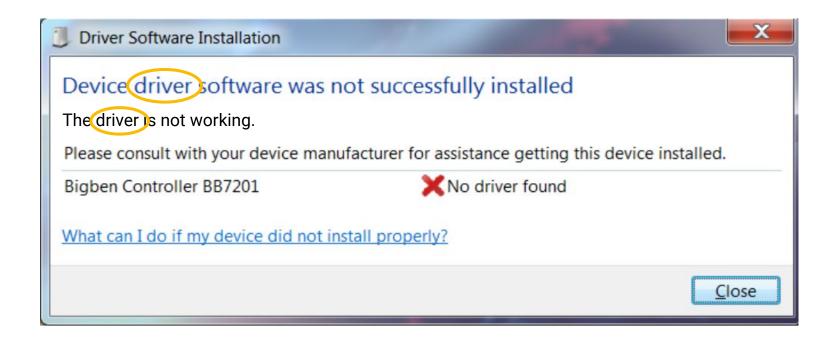
- Prepare train and test sets.
- 2 Use pre-built model.
- 3 Evaluate results.

#### **Custom Models**

- Prepare train and test sets.
- 2 Build model.
- ? Evaluate results.



### Why is domain-specific translation important?



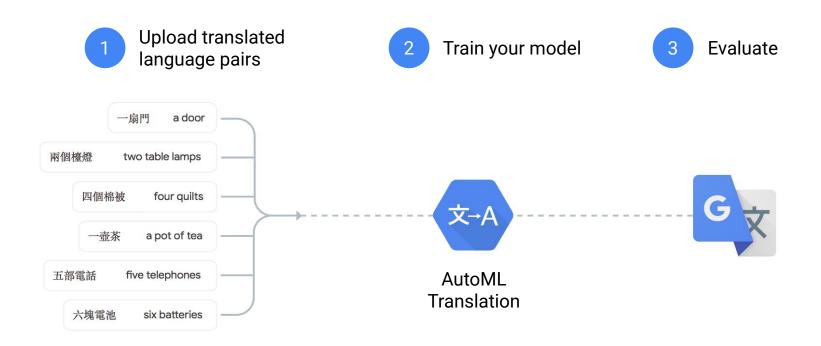


### What is the correct translation?





### How AutoML Translation<sup>BETA</sup> works?



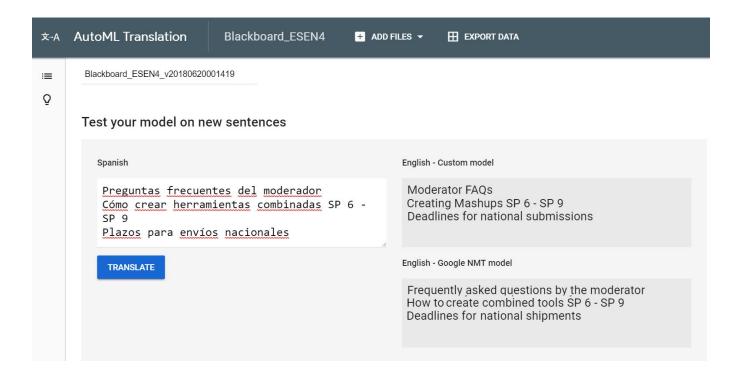


## How AutoML Translation<sup>BETA</sup> works?

t-A	AutoML Translation	blackboard_es_en   ■ ADD FILES -   ■ EXPORT DATA	next18-keynote-demo ▼
-	SENTENCES TRAIN	PREDICT	
Ô	blackboardESEN_te 2500 blackboardESENtr., 123265	blackboardESENtrain.tmx (TRAIN) Showing first 20 of 123265 pairs	
	blackboardESEN_tu 4147	Spanish	English
		Acceder a la página de la lista de envios	Access the Submission List Page
		Aquí le brindamos toda la información que necesita para comenzar a utilizar Bb Grader.	Here's what you need to know to get started with using Bb Grader.
		Cuando se visualiza un envío de grupo, los profesores pueden ver los miembros individuales si tocan la diapositiva Miembros del grupo que aparece al final del panel de calificación.	When viewing a group submission, instructors can view the individual members by tapping the Group Members slide out at the bottom of the grading panel.
		Se abrirá el panel de calificación.	This opens the grading panel.
		La barra de finalización situada debajo de cada actividad indica a los profesores cuántos alumnos aproximadamente han enviado la actividad.	Instructors can see approximately how many students have submitted the assignment based on the completion bar beneath the assignment.
		Puede grabar un video nuevo en directo o usar un video pregrabado de la bibliofeca fotográfica.	You can record a new, live video or use a pre-recorded video from your Photo Library.
		Categorias de encabezado	Heading Categories
		Todos	All:
		Aplicar calificaciones	Apply Grades
		Início rápido	Quick Start
		El comentario en video debe ser conciso.	Keep your video feedback concise.
		Para obtener más información sobre la calificación de envios, consulte Calificar envios.	To learn more about grading submissions, see Grade Submissions.
		Puede explorar los envios de carpeta con las flechas que hay en la parte superior del envio.	You can page through portfolio submissions by using the arrows at the top of the submission.
		Detalles de la actividad	Assignment Details
		Abra la aplicación y busque su institución, ya sea mediante la URL de la instancia de Learn de su institución o el nombre de su institución.	Open the app and search for your institution either by the UR for your institution's instance of Learn or your institution name.
		Toque un criterio para aplicar una puntuación.	Tap a rubric criterion to apply a score.
		Comentarios en vídeo	Video Feedback
		Las calificaciones individuales de un envio de grupo determinado se pueden ver en la página Lista de envios; para ello, deslice el dedo hacia la izquierda para mostrar el botón Perfil de grupo.	Individual Grades for a particular group's submission can be viewed on the Submission Listing page by swiping to the left to reveal the Group Profile button.
		Microsoft Word	Microsoft Word
		Solo se puede usar un criterio a la vez por cada envio.	Only one rubric may be used at a time per submission.



### **Education domain**





### Natural language conversations



An end-to-end developer platform for building natural and rich conversational experiences.



## There are three key components in Dialogflow

- Intents
- 2 Entities
- 3 Contexts



### Intents are the actions a user wants to take

2 lattes please Intent: Order coffee

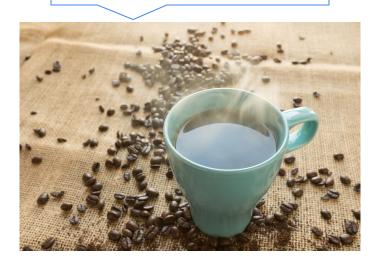






## Entities are the nouns in your dialog

I'd like an Americano!





### Context helps the chatbot keep track

What is the weather in **San Francisco**, **today**?



Beautiful and sunny! 70 degrees.

What about tomorrow?



## Lab

Getting started with Dialogflow

In this lab, we'll create a chatbox using Dialogflow.

## Lab Steps

- 1. Familiarize yourself with Dialogflow console.
- 2. Learn to create intents.
- 3. Extract data with entities.
- 4. Manage state with contexts.
- 5. Test the chatbot.

cloud.google.com

