Deep Learning
The Good, the Bad and the Ugly
Łukasz Kaiser
• Intro
• Basics
  • Tensor view of neural networks
  • TensorFlow core and higher-level APIs, Tensor2Tensor
• Exercise: train image classification models and explore them
• Sequence models
  • Deterministic models, expressive power and the Neural GPU
  • Autoregressive models, Transformer and Universal Transformer
• Exercise: train sequence models and explore them
• Outlook: deep learning, RL and community
But Why?
**Speed**

TPUv2: 180 TF/2$/h  
TPUv2 pod: 11.5 PF  
TPUv3 pod: over 100 PF  
Top supercomputer: 122 PF  
(Double precision, could be over 1 exaflop for ML applications.)
ML Arxiv Papers per Year

~50 New ML papers every day!
Rapid accuracy improvements

Image courtesy of Canziani et al, 2017
Radically open culture

James Bradbury
@jekbradbury

Replying to @seb_ruder

That's not the latest adaptive learning rate method any more 😅, the latest adaptive learning rate method is AdaFactor, quietly added three weeks ago to the Tensor2Tensor repository along with a note reading "TODO(noam): write a paper."

github.com/tensorflow/ten...
How Deep Learning Quietly Revolutionized NLP (2016)
What NLP tasks are we talking about?

- Part Of Speech Tagging: Assign part-of-speech to each word.
- Parsing: Create a grammar tree given a sentence.
- Named Entity Recognition: Recognize people, places, etc. in a sentence.
- Language Modeling: Generate natural sentences.
- Translation: Translate a sentence into another language.
- Sentence Compression: Remove words to summarize a sentence.
- Abstractive Summarization: Summarize a paragraph in new words.
- Question Answering: Answer a question, maybe given a passage.
- ....
Can deep learning solve these tasks?

- Inputs and outputs have variable size, how can neural networks handle it?
- **Recurrent Neural Networks** can do it, but how do we train them?
- **Long Short-Term Memory** [Hochreiter et al., 1997], but how to compose it?
- **Encoder-Decoder** (sequence-to-sequence) architectures [Sutskever et al., 2014; Bahdanau et al., 2014; Cho et al., 2014]
Parsing with sequence-to-sequence LSTMs

(1) Represent the tree as a sequence.

(2) Generate data and train a sequence-to-sequence LSTM model.

(3) Results: 92.8 F1 score vs 92.4 previous best [Vinyals & Kaiser et al., 2014]
### Language modeling with LSTMs

Language model performance is measured in **perplexity** (lower is better).

<table>
<thead>
<tr>
<th>Model Size</th>
<th>Perplexity</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kneser-Ney 5-gram:</td>
<td>67.6</td>
<td>[Chelba et al., 2013]</td>
</tr>
<tr>
<td>RNN-1024 + 9-gram:</td>
<td>51.3</td>
<td>[Chelba et al., 2013]</td>
</tr>
<tr>
<td>LSTM-512-512:</td>
<td>54.1</td>
<td>[Józefowicz et al., 2016]</td>
</tr>
<tr>
<td>2-layer LSTM-8192-1024:</td>
<td>30.6</td>
<td>[Józefowicz et al., 2016]</td>
</tr>
<tr>
<td>2-l.-LSTM-4096-1024+MoE:</td>
<td>28.0</td>
<td>[Shazeer &amp; Mirhoseini et al., 2016]</td>
</tr>
</tbody>
</table>

Model size seems to be the decisive factor.
About 800 people gathered at Hever Castle on Long Beach from noon to 2pm, three to four times that of the funeral cortege.

It is now known that coffee and cacao products can do no harm on the body.

Yuri Zhirkov was in attendance at the Stamford Bridge at the start of the second half but neither Drogba nor Malouda was able to push on through the Barcelona defence.
Sentence compression with LSTMs

Example:
Input:  State Sen. Stewart Greenleaf discusses his proposed human trafficking bill at Calvery Baptist Church in Willow Grove Thursday night.
Output: Stewart Greenleaf discusses his human trafficking bill.

Results:  
<table>
<thead>
<tr>
<th></th>
<th>readability</th>
<th>informativeness</th>
</tr>
</thead>
<tbody>
<tr>
<td>MIRA (previous best):</td>
<td>4.31</td>
<td>3.55</td>
</tr>
<tr>
<td>LSTM [Filippova et al., 2015]:</td>
<td>4.51</td>
<td>3.78</td>
</tr>
</tbody>
</table>
Translation with LSTMs

Translation performance is measured in **BLEU scores** (higher is better, EnDe):

- **Phrase-Based MT**: 20.7 [Durrani et al., 2014]
- **Early LSTM model**: 19.4 [Sébastien et al., 2015]
- **DeepAtt (large LSTM)**: 20.6 [Zhou et al., 2016]
- **GNMT (large LSTM)**: 24.9 [Wu et al., 2016]
- **GNMT+MoE**: 26.0 [Shazeer & Mirhoseini et al., 2016]

Again, model size and tuning seem to be the decisive factor.
German:
Probleme kann man niemals mit derselben Denkweise lösen, durch die sie entstanden sind.

PBMT Translate:
No problem can be solved from the same consciousness that they have arisen.

GNMT Translate:
Problems can never be solved with the same way of thinking that caused them.
## Translation with LSTMs: How good is it?

Google Translate production data, median score by human evaluation on the scale 0-6.  
[Wu et al., ‘16]

<table>
<thead>
<tr>
<th>Language Pair</th>
<th>PBMT</th>
<th>GNMT</th>
<th>Human</th>
<th>Relative Improvement</th>
</tr>
</thead>
<tbody>
<tr>
<td>English → Spanish</td>
<td>4.885</td>
<td>5.428</td>
<td>5.504</td>
<td>87%</td>
</tr>
<tr>
<td>English → French</td>
<td>4.932</td>
<td>5.295</td>
<td>5.496</td>
<td>64%</td>
</tr>
<tr>
<td>English → Chinese</td>
<td>4.035</td>
<td>4.594</td>
<td>4.987</td>
<td>58%</td>
</tr>
<tr>
<td>Spanish → English</td>
<td>4.872</td>
<td>5.187</td>
<td>5.372</td>
<td>63%</td>
</tr>
<tr>
<td>French → English</td>
<td>5.046</td>
<td>5.343</td>
<td>5.404</td>
<td>83%</td>
</tr>
<tr>
<td>Chinese → English</td>
<td>3.694</td>
<td>4.263</td>
<td>4.636</td>
<td>60%</td>
</tr>
</tbody>
</table>
That was 2016. Now.
### Attention: Machine Translation Results

<table>
<thead>
<tr>
<th>Model</th>
<th>BLEU</th>
<th>Training Cost (FLOPs)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>EN-DE</td>
<td>EN-FR</td>
</tr>
<tr>
<td></td>
<td>EN-DE</td>
<td>EN-FR</td>
</tr>
<tr>
<td>ByteNet [17]</td>
<td>23.75</td>
<td></td>
</tr>
<tr>
<td>Deep-Att + PosUnk [37]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GNMT + RL [36]</td>
<td>24.6</td>
<td>2.3 \times 10^{19}</td>
</tr>
<tr>
<td>ConvS2S [9]</td>
<td>25.16</td>
<td>9.6 \times 10^{18}</td>
</tr>
<tr>
<td>MoE [31]</td>
<td>26.03</td>
<td>2.0 \times 10^{19}</td>
</tr>
<tr>
<td>Deep-Att + PosUnk Ensemble [37]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GNMT + RL Ensemble [36]</td>
<td>26.30</td>
<td>1.8 \times 10^{20}</td>
</tr>
<tr>
<td>ConvS2S Ensemble [9]</td>
<td>26.36</td>
<td>7.7 \times 10^{19}</td>
</tr>
<tr>
<td>Transformer (base model)</td>
<td>27.3</td>
<td>3.3 \times 10^{18}</td>
</tr>
<tr>
<td>Transformer (big)</td>
<td>28.4</td>
<td>2.3 \times 10^{19}</td>
</tr>
<tr>
<td></td>
<td>29.7</td>
<td></td>
</tr>
</tbody>
</table>
Basics
Old School View

\[ f(\sum_{i=1}^{n} W_i X_i) \]
Convolutions

\[ U * s[x, y, i] = \sum_{u=-[k_w/2]}^{[k_w/2]} \sum_{v=-[k_h/2]}^{[k_h/2]} \sum_{c=1}^{m} s[x + u, y + v, c] \cdot U[u, v, c, i]. \]
Modern View

\[ h = f(Wx + B) \quad \text{[or } h = \text{conv}(W, x)\text{]} \]
\[ o = f(W'h + B') \]
\[ l = -\log_p(o = \text{true}) \]

\[ P \leftarrow lr \times \frac{dl}{dP} \quad \text{where } P = \{W, W', B, B'\} \]
So what do we need?

1. Operations like matmul, f done fast
2. Gradients symbolically for dl/dP
3. Specify W,W’ and keep track of them
4. Run it on a large scale

See this online course for a nice introduction:
https://www.coursera.org/learn/machine-learning
Core TF Model
Yet another dataflow system

Graph of *Nodes*, also called *Operations* or *ops*.
Yet another dataflow system with tensors

Edges are N-dimensional arrays: Tensors
Yet another dataflow system with state

'Biases' is a variable

Some ops compute gradients

−= updates biases

biases

... → Add → ...

learning rate

−=

Mul
Yet another dataflow system

Devices: Processes, Machines, GPUs, etc
What's not in the Core Model

- Anything about neural networks, machine learning, ...
- Anything about backpropagation, differentiation, ...
- Anything about gradient descent, parameter servers...

These are built by combining existing operations, or defining new operations.

Core system can be applied to other problems than machine learning.
Core TF API
API Families

Graph Construction
- Assemble a Graph of Operations.

Graph Execution
- Deploy and execute operations in a Graph.
import tensorflow as tf

# Create an operation.
hello = tf.constant("Hello, world!")
# Create a session.
sess = tf.Session()
# Execute that operation and print its result.
print(sess.run(hello))
Graph Construction

Library of predefined Ops
- Constant, Variables, Math ops, etc.

Functions to add Ops for common needs
- Gradients: Add Ops to compute derivatives.
- Training methods: Add Ops to update variables (SGD, Adagrad, etc.)

All operations are added to a global *Default Graph*.
Slightly more advanced calls let you control the Graph more precisely.
Op that holds state that persists across calls to `Run()`

```python
v = tf.get_variable('v', [4, 3])  # 4x3 matrix, float by default
```
Some Ops modify the Variable state: InitVariable, Assign, AssignSub, AssignAdd.

\[
\text{init} = v.\text{assign}(\text{tf.random_uniform}(\text{shape}=v.\text{shape}))
\]
Math Ops

A variety of Operations for linear algebra, convolutions, etc.

c = tf.constant(...)
w = tf.get_variable(...)b = tf.get_variable(...)y = tf.add(tf.matmul(c, w), b)

Overloaded Python operators help: y = tf.matmul(c, w) + b
Operations, plenty of them

Documentation at tensorflow.org

- **Array ops**
  - Concat
  - Slice
  - Reshape
  - ...

- **Math ops**
  - Linear algebra (MatMul, ...)
  - Component-wise ops (Mul, ...)
  - Reduction ops (Sum, ...)

- **Neural network ops**
  - Non-linearities (Relu, ...)
  - Convolutions (Conv2D, ...)
  - Pooling (AvgPool, ...)

- **...and many more**
  - Constants, Data flow, Control flow,
    Embedding, Initialization, I/O, Legacy
  - Input Layers, Logging, Random,
    Sparse, State, Summary, etc.
Graph Construction Helpers

- Gradients
- Optimizers
- Higher-Level APIs in core TF
- Higher-Level libraries outside core TF
Gradients

Given a loss, add Ops to compute gradients for Variables.
Gradients

```
# Generate gradients
tf.gradients(loss, [var0, var1])
```

![Diagram showing the computation graph with operations and variables](image)
Example

Gradients for MatMul

x → Transpose → MatMul → gw

gy → MatMul → gx

w → Transpose → MatMul → y
Optimizers

Apply gradients to Variables: SGD(var, grad, learning_rate)

Note: learning_rate is just output of an Op, it can easily be decayed.
Easily Add Optimizers

Builtin
  ● SGD, Adagrad, Momentum, Adam, …

Contributed
  ● LazyAdam, NAdam, YellowFin, Adafactor, …
Putting all together to train a Neural Net

Build a Graph by adding Operations:
- For **Variables** to hold the parameters of the Neural Net.
- To compute the **Neural Net output**: e.g. classification predictions.
- To compute a **training loss**: e.g. cross entropy, parameter L2 norms.
- To **calculate gradients** for the parameters to train.
- To **apply gradients** with a training function.
Distributed Execution
Graph Execution

Session API
- API to deploy a Graph in a Tensorflow runtime
- Can run any subset of the graph
- Can add Ops to an existing Graph (for interactive use in colab for example)

Training Utilities
- Checkpoint, Recovery, Summaries, Replicas, etc.
Local Runtime

Python Program
create graph
create session
sess.run()
Remote Runtime

Python Program
create graph
create session
sess.run()

Session

Master
RunSubGraph()
CreateGraph()
Run(ops)

Worker
Worker
Worker
Worker

CPU
CPU
CPU
CPU

GPU
GPU
GPU
GPU

GetTensor()
# Run an Op and fetch its output.
# "values" is a numpy ndarray.
values = sess.run(<an op output>)
Running and fetching output

Transitive closure of needed ops is Run
Execution happens in parallel
a_val = ...a numpy ndarray...
values = sess.run(<an op output>,
    feed_input({<a output>: a_val}))
Feeding input, Running, and Fetching

Only the required Ops are run.
Higher-Level Core TF API
def embedding(x, vocab_size, dense_size, 
              name=None, reuse=None, multiplier=1.0):
    """Embed x of type int64 into dense vectors."""
    with tf.variable_scope(  # Use scopes like this.
        name, default_name="emb", values=[x], reuse=reuse):
        embedding_var = tf.get_variable(
            "kernel", [vocab_size, dense_size])
        return tf.gather(embedding_var, x)
def bytenet(inputs, targets, hparams):
    final_encoder = common_layers.residual_dilated_conv(
        inputs, hparams.num_block_repeat, "SAME", "encoder", hparams)
    shifted_targets = common_layers.shift_left(targets)
    kernel = (hparams.kernel_height, hparams.kernel_width)
    decoder_start = common_layers.conv_block(
        tf.concat([final_encoder, shifted_targets], axis=3),
        hparams.hidden_size, [(1, 1), kernel], padding="LEFT")
    return common_layers.residual_dilated_conv(
        decoder_start, hparams.num_block_repeat,
        "LEFT", "decoder", hparams)
Training Utilities

Training program typically runs multiple threads

- Execute the training op in a loop.
- Checkpoint every so often.
- Gather summaries for the Visualizer.
- Other, eg. monitors Nans, costs, etc.
Estimator
So what do we need?

1. Operations like matmul, f done fast
2. Gradients symbolically for $\frac{dl}{dP}$
3. Specify $W, W'$ and keep track of them
4. Run it on a large scale

TensorFlow view:

\[
\begin{align*}
  h &= tf.layers.dense(x, h_{\text{size}}, \text{name}="h1") \\
  o &= tf.layers.dense(h, 1, \text{name}="output") \\
  l &= tf.nn.sparse_softmax_cross_entropy_with_logits(logits=o, \text{labels}=y)
\end{align*}
\]

But data? Where do we get \{x,y\} from?

\[
\begin{align*}
  h &= f(Wx + B) \\
  o &= f(W'h + B') \\
  l &= -\log p(o = y) \\
  P &= P - \text{lr} \times \frac{dl}{dP}
\end{align*}
\]
Tensor2Tensor (T2T) is a library of deep learning models and datasets designed to accelerate deep learning research and make it more accessible.

- **Datasets**: ImageNet, CIFAR, MNIST, Coco, WMT, LM1B, ...
- **Models**: ResNet, RevNet, ShakeShake, Xception, SliceNet, Transformer, ByteNet, Neural GPU, LSTM, ...
- **Tools**: cloud training, hyperparameter tuning, TPU, ...
So what do we need?

1. Operations like matmul, f done fast
2. Gradients symbolically for dl/dP
3. Specify W,W’ and keep track of them
4. Run it on a large scale

TensorFlow: goo.gl/WuPSzb

```python
x, y = mnist.dataset
h = tf.layers.dense(x, h_size, name="h1")
o = tf.layers.dense(h, 1, name="output")
l = tf.nn.sparse_softmax_cross_entropy_with_logits(logits=o, labels=y)
```

\[
h = f(Wx + B) \\
o = f(W'h + B') \\
l = -\log p(o = true) \\
P -= lr * \frac{dl}{dP}
\]
Play with the colab and add to goo.gl/zCgNdJ

goo.gl/WuPSzb

- Try pure SGD instead of the Adam optimizer and others like AdaFactor
  - Find in tensorflow.org where is the API and how optimizers are called
  - Find the AdaFactor paper on arxiv and read it; use it from Tensor2Tensor
- Try other layer sizes and numbers of layers, other activation functions.
- Try running a few times, how does initialization affect results?
- Try running on Cifar10, how does your model perform?
- Make a convolutional model, is it better? (tf.layers.dense -> tf.layers.conv2d)
- Try residual connections through conv layers, check out shake-shake in T2T
Sequence Models

(deterministic, theory-oriented)
What’s That?

Can you recognize these patterns?

Should we expect a computer to do so?

- All patterns? What are patterns?
- Simple computer? Advanced?
- Does it require a search?
- Can the training data have noise?
- In what time should we do it?
- Can a neural network learn it?
- Just with gradient descent?

**Pattern 1:**

i: 10 i: 11 i: 10110 i: 1000 i: 1101
o: 01 o: 11 o: 01101 o: 0001 o: 1011

**Pattern 2:**

i: 1 i: 10 i: 101 i: 101____ i: 1000____
o: 11 o: 1010 o: 101101 o: 10001000

**Pattern 3:**

i: 121 i: 10210 i: 01210 i: 01201 i: 11201
o: 1 i: 10210 i: 01210 i: 01201 i: 11201
o: 1 o: 101 o: 101 o: 011 o: 001 o: 011

i: 11211 i: 1012100 i: 1012010 i: 011020101
o: 101 o: 101 o: 0101 o: 00111
Gradient Descent?

Gradient descent is the standard way of training neural networks and many other forms of machine learning.

But it is a very basic method that only finds local minima. It is counter-intuitive to apply it to combinatorial problems. But very high-dimensional spaces are often counter-intuitive and proper representations can make gradient descent applicable to combinatorial problems too.
Computing with Neural Networks

Basic feed-forward neural networks operate on fixed-size vectors, so their power to generalize is limited.

Recurrent neural networks lift this major limitation, but their architecture restricts their power in other ways.

**Feed-forward neural network:**
- fixed-size input
- fixed-size output
- number of parameters depends on these sizes
- limited computational power

**Recurrent neural network:**
- variable-size input
- variable-size output
- number of parameters depends on memory size
- computational power limited by this memory size

Great success in speech recognition and language processing!
Why Learn Algorithms?

Algorithms are universal patterns, so if we can learn them, then in principle we can learn everything. If we can’t learn basic ones, then we can’t hope to build universal machine intelligence.

Can’t recurrent neural networks do it?
- Only if testing length ~ training.
- But with higher length we need to increase network capacity (number of parameters).
- Fail to capture algorithmic patterns, even in principle.

The big dream of universal learning.

Kolmogoroff complexity
Solomonoff induction
Universal AI

Previous work on neural algorithm learning.
- RNN and LSTM-based, generalizes a bit with attention.
- Data-structure based (stack RNN, memory nets).
- Neural Turing Machines.

Problems with Neural Turing Machines.
- Look close, your computer does not have a tape!
- Very sequential in nature, hard to train on hard problems.
- Example problem: long multiplication.
Recurrent Neural Networks share weights across time steps but use fixed-size memory vectors.

Time: $O(f(n))$  Space: $O(1)$

Neural Turing Machines work with variable-size memory and so can do general computation, but are sequential.

Time: $O(f(n))$  Space: $O(g(n))$

Step Op:

$$\tanh(xW+B)$$

$$\ldots$$
Neural computation requires operators that use a **fixed number of parameters** but operate on **memory of arbitrary size**. There are a few candidates:

1) Attention (Neural Turing Machine)
2) Stack, (De)Queue, Lists
3) ... other memory structures ...
4) Convolutions!

**Why convolutions?** They act like a neural GPU.

- Attention is a softmax, effectively $\sim 1$ memory item / step.
- Similarly a stack and other task-specific memory structures.
- Convolutions affect all memory items in each step.
- Convolutions are already implemented and optimized.
- To train well we need to use LSTM-style gates: **CGRN**.

**Attention mechanism:**

$$c \cdot W + m \cdot W' = x_i$$

$$\text{softmax}(v^T \cdot \tanh(x_i))$$
Convolutional Gated Recurrent Networks (CGRNs) perform many parallel operations in each step, akin to a neural GPU and in contrast to the sequential nature of Neural Turing Machines.

The definition of a CGRN is also very simple and elegant, see on the right!

**CGRU definition:** (similar to GRU replacing linear by convolution)

\[
s_{t+1} = g \cdot s_t + (1 - g) \cdot c \quad \text{where:}
\]

\[
g = \text{sigmoid}(\text{conv}(s_t, K_g))
\]

\[
c = \text{tanh}(\text{conv}(s_t \cdot r, K_c))
\]

\[
r = \text{sigmoid}(\text{conv}(s_t, K_r))
\]

**Computational power:**

- Small number of parameters (3 kernels: \(K_g\), \(K_c\), \(K_r\)).
- Can simulate computation of cellular automata.
- With memory of size \(n\) can do \(n\) local operations / step.
- E.g., can do long multiplication in \(O(n)\) steps.
Neural GPU

Convolutional Gated Recurrent Networks (CGRNs) perform many parallel operations in each step, akin to a neural GPU or a cellular automaton.

**CGRU definition:** (similar to GRU replacing linear by convolution)

\[ m_{t+1} = g \cdot m_t + (1 - g) \cdot c \]

where:

\[ g = \text{sigmoid}(\text{conv}(m_t, K_g)) \]

\[ c = \text{tanh}(\text{conv}(m_t \cdot r, K_c)) \]

\[ r = \text{sigmoid}(\text{conv}(m_t, K_r)) \]
Neural GPU Results

A n-step n-size memory CGRN achieves **100x generalization** on a number of tasks such as reversing sequences, long addition, or even non-linear-time **long multiplication**.

We say an output is correct only if all its digits are right. So an error of 40% means that there are still 60% of fully correct sequences on output.

### Reversing:
- i: 3 0 8 4
- o: 4 8 0 3

(all examples for length=4)

### Long decimal addition:
- i: 4 5 3 0 OP + 0 9 9 5
- o: 4 4 3 6

(OP+, OP- are input characters)

### Long binary multiplication:
- i: 0 1 1 0 OP * 0 1 0 1
- o: 0 0 1 1 1

(6 * 10 = 60 = 32+16+8+4)

* Training upto length 20.

<table>
<thead>
<tr>
<th>Length</th>
<th>Neural GPU Error</th>
<th>LSTM+A Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>10*</td>
<td>0%</td>
<td>0%</td>
</tr>
<tr>
<td>20*</td>
<td>0%</td>
<td>0%</td>
</tr>
<tr>
<td>60</td>
<td>0%</td>
<td>7%</td>
</tr>
<tr>
<td>10*</td>
<td>0%</td>
<td>34%</td>
</tr>
<tr>
<td>20*</td>
<td>0%</td>
<td>100%</td>
</tr>
<tr>
<td>25</td>
<td>0%</td>
<td>100%</td>
</tr>
<tr>
<td>2000</td>
<td>0%</td>
<td>100%</td>
</tr>
<tr>
<td>10*</td>
<td>0%</td>
<td>91%</td>
</tr>
<tr>
<td>20*</td>
<td>0%</td>
<td>100%</td>
</tr>
<tr>
<td>25</td>
<td>0%</td>
<td>100%</td>
</tr>
<tr>
<td>2000</td>
<td>0%</td>
<td>100%</td>
</tr>
</tbody>
</table>
A number of techniques are needed to make the training of a Neural GPU work well, and some are required for the generalization to work or to be stable.

**Curriculum learning.**
- Start training on small lengths, increase when learned.

**Parameter sharing relaxation.**
- Allow to do different things in different time-steps first.
- Not needed now with bigger models and orthogonal init.

**Dropout on recurrent connections.**
- Randomly set 10% of state vectors to 0 in each step.
- Interestingly, this is key for good length generalization.

**Noise added to gradients.**
- Add small gaussian noise to gradients in each training step.

**Gate cutoff (saturation, probably not needed with other).**
- Instead of sigmoid(x) use \[1.2 \text{sigmoid}(x) - 0.1\] \([0, 1]\).

**Tune parameters. Tune, tune, tune.**
- A lot of GPUs running for a few months; or: better method!
Play with the colab and add to goo.gl/zCgNdJ

goo.gl/WuPSzb

- Try single-hidden-layer conv on “repeat previous digit” task
  - Problems with positions? How to handle variable-length sequences?
  - Try replacing fixed positional information with gating.
  - See that it works but fails on “repeat -3 digit” with kernel 3
- Try a recurrent model in the colab, see that “-3 digit” works now
- For ambitious: try to replicate the Neural GPU:
  - Best to start with the Improved Neural GPU: find paper and read
  - https://github.com/tensorflow/tensor2tensor/blob/master/tensor2tensor/models/neural_gpu.py#L70
  - You’ll want to borrow AdaMax from the Improved Neural GPU repo
Sequence Models

(autoregressive, practice-oriented)
What’s wrong with previous one?

1. The world is not deterministic
   ○ In case of an output distribution, it breaks
   ○ But autoregressive prediction can be used!
2. It is slow
   ○ Convolutions have limited field of view: attention.
   ○ We should not always do N steps: adaptive recurrence.
3. We need to do real-world sequence tasks
   ○ Translation, NLP, speech recognition, image generation.
   ○ Can we train on really long sequences?
RNNs Everywhere

Sequence to Sequence Learning with Neural Networks
Auto-Regressive CNNs

**WaveNet and ByteNet**

<table>
<thead>
<tr>
<th>Input</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1" alt="Input Layer" /></td>
<td><img src="image2" alt="Output Layer" /></td>
</tr>
<tr>
<td><img src="image3" alt="Hidden Layer" /></td>
<td><img src="image4" alt="Hidden Layer" /></td>
</tr>
</tbody>
</table>
Transformer

Attention

Convolution

Attention
Dot-Product Attention

\[ A(Q, K, V) = \text{softmax}(QK^T)V \]
Dot-Product Attention

\[ A(Q, K, V) = softmax(QK^T)V \]

def dot_product_attention(q, k, v, bias, dropout_rate=0.0, image_shapes=None, name=None,
                        make_image_summary=True, save_weights_to=None, dropout_broadcast_dims=None):

    with tf.variable_scope(
        name, default_name="dot_product_attention", values=[q, k, v]) as scope:
        # [batch, num_heads, query_length, memory_length]
        logits = tf.matmul(q, k, transpose_b=True)
        if bias is not None:
            logits += bias
        weights = tf.nn.softmax(logits, name="attention_weights")
        if save_weights_to is not None:
            save_weights_to[scope.name] = weights
        # dropping out the attention links for each of the heads
        weights = common_layers.dropout_with_broadcast_dims(
            weights, 1.0 - dropout_rate, broadcast_dims=dropout_broadcast_dims)
        if expert_utils.should_generate_summaries() and make_image_summary:
            attention_image_summary(weights, image_shapes)
        return tf.matmul(weights, v)
<table>
<thead>
<tr>
<th></th>
<th>Ops</th>
<th>Activations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attention (dot-prod)</td>
<td>$n^2 \cdot d$</td>
<td>$n^2 + n \cdot d$</td>
</tr>
<tr>
<td>Attention (additive)</td>
<td>$n^2 \cdot d$</td>
<td>$n^2 \cdot d$</td>
</tr>
<tr>
<td>Recurrent</td>
<td>$n \cdot d^2$</td>
<td>$n \cdot d$</td>
</tr>
<tr>
<td>Convolutional</td>
<td>$n \cdot d^2$</td>
<td>$n \cdot d$</td>
</tr>
</tbody>
</table>

$n = \text{sequence length}$  
$d = \text{depth}$  
k = \text{kernel size}$
What’s missing from Self-Attention?

Convolution

Self-Attention
What’s missing from Self-Attention?

- Convolution: a different linear transformation for each relative position. Allows you to distinguish what information came from where.

- Self-Attention: a weighted average :(

Convolution

Self-Attention
The Fix: Multi-Head Attention

- Multiple attention layers (heads) in parallel (shown by different colors).
- Each head uses different linear transformations.
- Different heads can learn different relationships.
The Fix: Multi-Head Attention
The Fix: Multi-Head Attention
<table>
<thead>
<tr>
<th></th>
<th>Ops</th>
<th>Activations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Multi-Head Attention with linear transformations. For each of the h heads, (d_q = d_k = d_v = d/h)</td>
<td>(n^2 \cdot d + n \cdot d^2)</td>
<td>(n^2 \cdot h + n \cdot d)</td>
</tr>
<tr>
<td>Recurrent</td>
<td>(n \cdot d^2)</td>
<td>(n \cdot d)</td>
</tr>
<tr>
<td>Convolutional</td>
<td>(n \cdot d^2)</td>
<td>(n \cdot d)</td>
</tr>
</tbody>
</table>

\(n = \text{sequence length} \quad d = \text{depth} \quad k = \text{kernel size}\)
Three ways of attention

Encoder-Decoder Attention

Encoder Self-Attention

MaskedDecoder Self-Attention
The Transformer

Figure 1: The Transformer - model architecture.
### Machine Translation Results: WMT-14

<table>
<thead>
<tr>
<th>Model</th>
<th>BLEU</th>
<th>Training Cost (FLOPs)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>EN-DE</td>
<td>EN-FR</td>
</tr>
<tr>
<td>ByteNet [17]</td>
<td>23.75</td>
<td>39.2</td>
</tr>
<tr>
<td>Deep-Att + PosUnk [37]</td>
<td>24.6</td>
<td>39.92</td>
</tr>
<tr>
<td>GNMT + RL [36]</td>
<td>25.16</td>
<td>40.46</td>
</tr>
<tr>
<td>ConvS2S [9]</td>
<td>26.03</td>
<td>40.56</td>
</tr>
<tr>
<td>MoE [31]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Deep-Att + PosUnk Ensemble [37]</td>
<td></td>
<td>40.4</td>
</tr>
<tr>
<td>GNMT + RL Ensemble [36]</td>
<td>26.30</td>
<td>41.16</td>
</tr>
<tr>
<td>ConvS2S Ensemble [9]</td>
<td>26.36</td>
<td>41.29</td>
</tr>
<tr>
<td>Transformer (base model)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Transformer (big)</td>
<td>28.4</td>
<td>41.0</td>
</tr>
</tbody>
</table>
## Ablations

<table>
<thead>
<tr>
<th></th>
<th>$N$</th>
<th>$d_{\text{model}}$</th>
<th>$d_{\text{ff}}$</th>
<th>$h$</th>
<th>$d_k$</th>
<th>$d_v$</th>
<th>$P_{\text{drop}}$</th>
<th>$\epsilon_{ls}$</th>
<th>train steps</th>
<th>PPL (dev)</th>
<th>BLEU (dev)</th>
<th>params $\times 10^6$</th>
</tr>
</thead>
<tbody>
<tr>
<td>base</td>
<td>6</td>
<td>512</td>
<td>2048</td>
<td>8</td>
<td>64</td>
<td>64</td>
<td>0.1</td>
<td>0.1</td>
<td>100K</td>
<td>4.92</td>
<td>25.8</td>
<td>65</td>
</tr>
<tr>
<td>(A)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>512</td>
<td>512</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>5.29</td>
<td>24.9</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>128</td>
<td>128</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>5.00</td>
<td>25.5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>16</td>
<td>32</td>
<td>32</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>4.91</td>
<td>25.8</td>
<td></td>
</tr>
<tr>
<td></td>
<td>32</td>
<td>16</td>
<td>16</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>5.01</td>
<td>25.4</td>
<td></td>
</tr>
<tr>
<td>(B)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>16</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>5.16</td>
<td>25.1</td>
<td>58</td>
</tr>
<tr>
<td></td>
<td></td>
<td>32</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>5.01</td>
<td>25.4</td>
<td>60</td>
</tr>
<tr>
<td>(C)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>6.11</td>
<td>23.7</td>
<td>36</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>5.19</td>
<td>25.3</td>
<td>50</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>4.88</td>
<td>25.5</td>
<td>80</td>
</tr>
<tr>
<td></td>
<td>256</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>5.75</td>
<td>24.5</td>
<td>28</td>
</tr>
<tr>
<td></td>
<td>1024</td>
<td>32</td>
<td>32</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>4.66</td>
<td>26.0</td>
<td>168</td>
</tr>
<tr>
<td></td>
<td>1024</td>
<td>128</td>
<td>128</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>5.12</td>
<td>25.4</td>
<td>53</td>
</tr>
<tr>
<td></td>
<td>4096</td>
<td>1024</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>4.75</td>
<td>26.2</td>
<td>90</td>
</tr>
<tr>
<td>(D)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>5.77</td>
<td>24.6</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>4.95</td>
<td>25.5</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>4.67</td>
<td>25.3</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>5.47</td>
<td>25.7</td>
<td></td>
</tr>
<tr>
<td>(E)</td>
<td></td>
<td>positional embedding instead of sinusoids</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>big</td>
<td>6</td>
<td>1024</td>
<td>4096</td>
<td>16</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>4.33</td>
<td>26.4</td>
<td>213</td>
</tr>
</tbody>
</table>
Coreference resolution (Winograd schemas)
### Coreference resolution (Winograd schemas)

<table>
<thead>
<tr>
<th>Sentence</th>
<th>Google Translate</th>
<th>Transformer</th>
</tr>
</thead>
<tbody>
<tr>
<td>The cow ate the hay because it was delicious.</td>
<td>La vache mangeait le foin parce qu'elle était délicieuse.</td>
<td>La vache a mangé le foin parce qu'il était délicieux.</td>
</tr>
<tr>
<td>The cow ate the hay because it was hungry.</td>
<td>La vache mangeait le foin parce qu'elle avait faim.</td>
<td>La vache mangeait le foin parce qu'elle avait faim.</td>
</tr>
<tr>
<td>The women stopped drinking the wines because they were carcinogenic.</td>
<td>Les femmes ont cessé de boire les vins parce qu'ils étaient cancérogènes.</td>
<td>Les femmes ont cessé de boire les vins parce qu'ils étaient cancérigènes.</td>
</tr>
<tr>
<td>The women stopped drinking the wines because they were pregnant.</td>
<td>Les femmes ont cessé de boire les vins parce qu'ils étaient enceintes.</td>
<td>Les femmes ont cessé de boire les vins parce qu'elles étaient enceintes.</td>
</tr>
<tr>
<td>The city councilmen refused the female demonstrators a permit because they advocated violence.</td>
<td>Les conseillers municipaux ont refusé aux femmes manifestantes un permis parce qu'ils préconisaient la violence.</td>
<td>Le conseil municipal a refusé aux manifestantes un permis parce qu'elles prônaient la violence.</td>
</tr>
<tr>
<td>The city councilmen refused the female demonstrators a permit because they feared violence.</td>
<td>Les conseillers municipaux ont refusé aux femmes manifestantes un permis parce qu'ils craignaient la violence.</td>
<td>Le conseil municipal a refusé aux manifestantes un permis parce qu'elles craignaient la violence.*</td>
</tr>
</tbody>
</table>
Long Text Generation

Generating entire Wikipedia articles by summarizing top search results and references.

(Memory-Compressed Attn.)
"The Transformer" are a Japanese [[hardcore punk]] band.

==Early years==

The band was formed in 1968, during the height of Japanese music history. Among the legendary [[Japanese people|Japanese]] composers of [Japanese lyrics], they prominently exemplified Motohiro Oda's especially tasty lyrics and psychedelic intention. Michio was a longtime member of the every Sunday night band PSM. His alluring was of such importance as being the man who ignored the already successful image and that he municipal makeup whose parents were-- the band was called Jenei.&lt;ref&gt;[http://www.separatist.org/se_frontend/post-punk-musician-the-kidney.html](http://www.separatist.org/se_frontend/post-punk-musician-the-kidney.html)&lt;/ref&gt;

From a young age the band was very close, thus opting to pioneer what
From a young age the band was very close, thus opting to pioneer what
had actually begun as a more manageable core hardcore punk
band.&lt;ref&gt;http://www.talkradio.net/article/independent-music-fades-from-the-closed-drawings-out&lt;/ref&gt;

==History==

===Born from the heavy metal revolution===

In 1977 the self-proclaimed King of Tesponsors, [Joe Lus:

: It was somewhere... it was just a guile ... taking this song to
Broadway. It was the first record I ever heard on A.M., After some
opposition I received at the hands of Parsons, and in the follow-up
notes myself.&lt;ref&gt;http://www.discogs.com/artist/The+Op%C5%8Dn+%+Psalm&lt;/ref&gt;

The band cut their first record album titled "Transformed, furthered
The band cut their first record album titled "Transformed, furthered
and extended Extended", [https://www.discogs.com/album/69771
MC – Transformed EP (CDR) by The Moondrawn – EMI, 1994] and in 1978 the official band line-up of the three-piece pop-punk-rock
band TEEM. They generally played around [[Japan]], growing from the
Top 40 standard.

1981-2010: The band to break away

On 1 January 1981 bassist Michio Kono, and the members of the original
line-up emerged. Niji Fukune and his [[Head poet|Head]] band (now
guitarist) Kazuya Kouda left the band in the hands of the band at the
In June 1987, Kono joined the band as a full-time drummer, playing a
few nights in a 4 or 5 hour stint with [[D-beat]]. Kono played through
the mid-1950s, at Shinlie, continued to play concerts with drummers in
Ibis, Cor, and a few at the Leo Somu Studio in Japan. In 1987, Kono
recruited new bassist Michio Kono and drummer Ayaka Kurobe as drummer
for band. Kono played trumpet with supplement music with Saint Etienne
as a drummer. Over the next few years Kono played as drummer and would
get many alumni news invitations to the bands' "Toys Beach" section.
In 1999 he joined the [[CT-182]].

His successor was Barrie Bell on a cover of [[Jethro Tull
(band)|Jethro Tull]]'s original 1967 hit "Back Home" (last appearance was in Jethro), with whom he shares a name.

===2010 – present: The band to split===

In 2006 the band split up and the remaining members reformed under the
name Starmirror, with Kono in tears, ....
"The Transformer" is a [book] by British [illuminatist] [Herman Muirhead], set in a post-apocalyptic world that border on a mysterious alien known as the "Transformer Planet" which is his trademark to save Earth. The book is about 25 years old, and it contains forty-one different demographic models of the human race, as in the cases of two fictional "groups", "Robtobeau" and "Richard" and "The Transformers Planet".

== Summary ==

The book benefits on the [3-D film], taking his one-third of the world's pure and gas age from 30 to 70 within its confines.

The book covers the world of the world of [Area 51] from around the worlds of Earth. It is judged by the ability of [telepathy] and [television], and provides color, line, and end-to-end observational work.
and end-to-end observational work.

To make the book up and document the recoverable quantum states of the universe, in order to inspire a generation that fantasy producing a tele-recording-offering machine is ideal. To make portions of this universe home, he recreates the rostrum obstacle-oriented framework Minou.&lt;/ref&gt;http://www.rewunting.net/voir/BestatNew/2007/press/Story.html)&lt;/ref&gt;

== "The Transformer"==

The book was the first on a [[Random Access Album|re-issue]] since its original version of "[[Robtobeau]]", despite the band naming itself a &quot;Transformer Planet&quot; in the book.&lt;/ref
name=prweb-the-1985&gt;{{cite

Today, &quot;[[The Transformers Planet]]&quot; is played entirely open-ended, there are more than just the four previously separate only bands. A number of its groups will live on one abandoned volcano in North America,
Conceptual "The Transformer" universe

Principals a setting-man named "The Supercongo Planet," who is a naturalistic device transferring voice and humour from "The Transformer Planet," whose two vice-maks appear often in this universe existence, and what the project in general are trying to highlight many societal institutions. Because of the way that the corporation has made it, loneliness, confidence, research and renting out these universes are difficult to organise without the bands creating their own universe. The scientist is none other than a singer and musician. Power plants are not only problematic, but if they want programmed them to create and perform the world's first Broadcast of itself once the universe started, but deliberately Acta Biological Station, db.us and BB on "The Transformer Planet", "The Transformer Planet", aren't other things Scheduled for.
A man called Dick Latanii Bartow, known the
greatest radio dot Wonderland administrator at influential arrangers
in a craze over the complex World of Biological Predacial Engineer in
Rodel bringing Earth into a 'sortjob' with fans. During this
'Socpurportedly Human', Conspiracy was being released to the world as
Baron Maadia on planet Nature. A world-renowned scientist named Julia
Samur is able to cosmouncish society and run for it - except us who is
he and he is before talking this entire T100 before Cell physiologist
Cygnets. Also, the hypnotic Mr. Mattei arrived, so it is Mischief who
over-manages for himself - but a rising duplicate of Phil Rideout
makes it almost affable. There is plenty of people at work to make
use of it and animal allies out of politics. But Someday in 1964, when
we were around, we were steadfast against the one man's machine and he
did an amazing job at the toe of the mysterious...
Mr. Suki who is an engineering desk lecturer at the University of

.............
Image Generation

<table>
<thead>
<tr>
<th>Model Type</th>
<th>% unrecognized (max = 50%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ResNet</td>
<td>4.0%</td>
</tr>
<tr>
<td>Superresolution GAN (Garcia’16)</td>
<td>8.5%</td>
</tr>
<tr>
<td>PixelRecursive (Dahl et al., 2017)</td>
<td>11%</td>
</tr>
<tr>
<td>Image Transformer</td>
<td>36.9%</td>
</tr>
</tbody>
</table>
How about GANs?
(Are GANs Created Equal? A Large-Scale Study)

Problem 1: Variance

Problem 2: Even best models are not great:

Image Transformer: 36.6
Play with the colab

goo.gl/WuPSzb

- Try a pre-trained Transformer on translation, see attentions.
- See https://jalammar.github.io/illustrated-transformer/
- Add Transformer layer on the previous sequence tasks, try it.
- Try the non-deterministic sequence task: 50% copy / 50% repeat-even:
  - See that previous sequence model fails on unclear outputs
  - Add auto-regressive part and attention
  - See that the new model is 50% correct (best possible)
  - *Does it generalize less with attention? Why? What could be done?
Universal Transformer

Joint work with Mostafa Dehghani, Stephan Gouws, Oriol Vinyals, Jakob Uszkoreit.
What’s wrong with previous one?

1. The world is not deterministic Done
   ○ In case of an output distribution, it breaks
   ○ But autoregressive prediction can be used!

2. It is slow
   ○ Convolutions have limited field of view: attention. Done
   ○ We should not always do N steps: adaptive recurrence. Missing

3. We need to do real-world sequence tasks Done
   ○ Translation, NLP, speech recognition, image generation.
   ○ Can we train on really long sequences?

Transformer is great but not computationally universal.
Transformer meets Neural GPU

Transformer + convolution layers repeated recurrently
+ adaptive computation time for speed

show model and results (link to be added later)
Outlook
Multiple Modalities

(One Model to Learn Them All)
Multiple Modalities: MultiModel
MultiModel: 4 WMT, ImageNet, COCO, WSJ, PTB

“A man that is sitting in front of a suitcase”

Category 127 (Male Human)

“Last week, Kigali raised the possibility of military retaliation after shells…”

“Can you give our readers some details on this?”

The above represents a triumph of either apathy or civility

To English

To Category

To French

To German

To Parse

“La semaine dernière, Kigali a soulevé la possibilité de représailles militaires après avoir débarqué des coquilles…”

“Können Sie unseren Lesern einige Details dazu geben?”

“S NP DT JJS /NP VP VBJ NP NP DT NN /NP PP IN NP NP NN /NP CC NP NN /NP /NP /PP /NP /VP . /S”
Multi-Task Learning

- OpenAI blog
  - blog.openai.com

- Salesforce paper

More will be coming!
Reinforcement Learning
Reinforcement Learning

- Sequence learning with sparse loss, variance
- Great success in many domains:
  - Atari games, Go, robotics, ...
  - Dota, [OpenAI five](https://sites.google.com/corp/view/modelbasedrlatari/home)
  - AutoML and learning to learn
- Data hungry and sometimes hard to explore
  - Sequence models of the world to rescue!
  - [https://sites.google.com/corp/view/modelbasedrlatari/home](https://sites.google.com/corp/view/modelbasedrlatari/home)
How do I get it?
Tensor2Tensor
Tensor2Tensor (T2T) is a library of deep learning models and datasets designed to make deep learning more accessible and accelerate ML research.

- **Datasets**: ImageNet, CIFAR, MNIST, Coco, WMT, LM1B, ...
- **Models**: ResNet, RevNet, ShakeShake, Xception, SliceNet, Transformer, ByteNet, Neural GPU, LSTM, ...
- **Tools**: cloud training, hyperparameter tuning, TPU...
That's not the latest adaptive learning rate method any more 😏, the latest adaptive learning rate method is AdaFactor, quietly added three weeks ago to the Tensor2Tensor repository along with a note reading "TODO(noam): write a paper."
Tensor2Tensor Code (github)

- data_generators/: datasets, must subclass Problem
- models/: models, must subclass T2TModel
- utils/, bin/, etc.: utilities, binaries, cloud helpers, ...

```bash
pip install tensor2tensor && t2t-trainer \
   --generate_data --data_dir=~/.t2t_data --output_dir=~/.t2t_train/mnist \
   --problems=image_mnist --model=shake_shake --hparams_set=shake_shake_quick \
   --train_steps=1000 --eval_steps=100
```
Tensor2Tensor Applications

pip install tensor2tensor && t2t-trainer \
   --generate_data --data_dir=~/.t2t_data --output_dir=~/.t2t_train/dir \
   --problems=$P --model=$M --hparams_set=$H

- Translation (state-of-the-art both on speed and accuracy):
  $P=translate_ende_wmt32k, \ M=transformer, \ H=transformer_big

- Image classification (CIFAR, also ImageNet):
  $P=image_cifar10, \ M=shake_shake, \ H=shakeshake_big

- Summarization (CNN):
  $P=summarize_cnn_dailymail32k, \ M=transformer, \ H=transformer_prepend

- Speech recognition (Librispeech):
  $P=librispeech, \ M=transformer, \ H=transformer_librispeech
Why Tensor2Tensor?

- No need to reinvent ML. Best practices and SOTA models.
- Modularity helps. Easy to change models, hparams, data.
- Trains everywhere. Multi-GPU, distributed, Cloud, TPUs.
- Used by Google Brain. Papers, preferred for Cloud TPU LMs.
- Great active community! Find us on github, gitter, groups, ...
Tensor2Tensor + CloudML

How do I train a model on my data?
See the Cloud ML poetry tutorial!

- How to hook up your data to the library of models.
- How to easily run on Cloud ML and use all features.
- How to tune the configuration of a model automatically

Result: Even with 20K data examples it generates poetry!
What Next?
Pescadores Weekly Inc (organized in 1978 as the Pescadores Weekly) is the official Weekly organizing body of science. It was established in 1978 atop the city of the center of the City of Gravity. In 1978 it became the United States National Museum of Science and Design. The growing Berklee Plant represents the physical community of typical worldwide plants.

Pescadores is named after the lord of the University of the Gravity of Drama, an interest he established in 1989 of the West Bird Plant. In the fall of 1978 the Faxophonius Institute, with the Today Natural History Foundation, published by the University of Georgia, and Anubis (Yale University).

==History==
Pescadores provided classic books on science and other philosophical sciences such as geoscience that accomplished science-fiction in Europe during 1976–1977, being the first university campus of the University of New Zealand. …

The "Pescadores Weekly" is the only daily newspaper in the Fort Hood metropolitan area, although the entire population of Fort Hood is majority white in population. Due to its not-for-profit focus as a newspaper, Carehead reporters and Hometime have expressed the need for a feature to focus on Fort Hood and its inner-city population.

According to the Pescadores Weekly Editors' annual survey, the "Pescadores Weekly" circulation was the largest in Texas, surpassing the "Evening Star Sans Monthly" made in 1992. By contrast, Terry Truehead's "The Best of Fort Hood" daily was the fourth-fastest in Texas (behind rival "AfterEllena" and "Expensive Ellena", topping the chart thirteen years in a row) and expanded by two thirds during the 2002-2003 Edition.

==Political affiliation==
The newspaper endorsed John McCain's majoritarian, Jack Abramoff, at the 2004 GOP convention.
Let’s Go Bigger!

- 23 Million Parameters
  - Utter Nonsense

- 340 Million Parameters
  - Often makes sense
  - Lots of world knowledge

- 6 Billion Parameters
  - ?

- 100 Billion Parameters
  - ?
Check it out now:  goo.gl/WuPSzb

- Based on https://github.com/tensorflow/tensor2tensor
- Easy to use data-sets and models code on github, tutorials, ...
- Community: external contributors (many 100+ LOC), 1k+ forks, ...
- SOTA on NLP (translation, lm, summarization), image classification, ...
- Train on Cloud ML and Cloud TPUs, tested pretrained models
- https://sites.google.com/corp/view/modelbasedrlatari/home

In the future:

- How will realistic text, image and video generation change society?
- Will large-scale multi-task models show more general intelligence?
Homeworks

goo.gl/292FY4