From Honey Bee to Mouse Brain

M4: Massively Multilingual, Massive Machine Translation

Aditya Siddhant, Ali Dabirmoghaddam, Ankur Bapna, Colin Cherry, Dmitry Lepikhin, George Foster, Isaac Caswell, James Kuczmarski, Kun Zhang, Macduff Hughes, Mahdis Mahdieh, Manisha Jain, Markus Freitag, Maxim Krikun, Melvin Johnson, Mia Chen, Naveen Arivazhagan, Orhan Firat, Roee Aharoni, Sébastien Jean, Sneha Kudugunta, Thang Luong, Wei Wang, Wolfgang Macherey, Yanping Huang, Yonghui Wu, Yuan Cao, Zhifeng Chen
1. Goal & Motivations
2. Project Phases
3. Lessons Learned
Our goal

Develop a universal machine translation model (i.e. one model for all languages and domains)

“Perhaps the way [of translation] is to descend, from each language, down to the common base of human communication -- the real but as yet undiscovered universal language -- and then re-emerge by whatever particular route is convenient.”

Warren Weaver (1949)
Motivation 1: Improve translation quality for all language pairs

Data distribution over language pairs

High-resource languages
- {Spanish, French, German, ...}

Low-resource languages
- {Yoruba, Sindhi, Hawaiian, ...}

Approaching human quality (> 100M examples)

Often not usable (< 1M examples)

25+ Billion Training Examples
**Motivation 2:**

Expand language coverage

---

**In the world, there are...**

<table>
<thead>
<tr>
<th>Total languages</th>
<th>African languages</th>
<th>Native Am. languages¹</th>
</tr>
</thead>
<tbody>
<tr>
<td>7,000+</td>
<td>2,000+</td>
<td>700+</td>
</tr>
</tbody>
</table>

**But Translate only supports...**

<table>
<thead>
<tr>
<th>Total languages</th>
<th>African languages</th>
<th>Native Am. languages</th>
</tr>
</thead>
<tbody>
<tr>
<td>103</td>
<td>11</td>
<td>0</td>
</tr>
</tbody>
</table>

---

1. Estimate is 766 Native Am languages globally: 383 in South America (source); 176 in central America (source); 207 in North America (source)
100 billion neurons

Neurons in the Human Brain

According to many estimates, the human brain contains around 100 billion neurons (give or take a few billion). June 11, 2019

How Many Neurons Are in the Brain? - Verywell Health

Each individual neuron can form thousands of links with other neurons in this way, giving a typical brain well over 100 trillion synapses (up to 1,000 trillion, by some estimates). Functionally related neurons connect to each other to form neural networks (also known as neural nets or assemblies).

Neurons & Synapses - Memory & the Brain - The Human Memory

www.human-memory.net/brain_neurons.html
Number of Synapses

Fruit fly
Honey bee
Mouse
Cat
Macaque
Human

$10^6 > 10^9 > 10^{12} > 10^{13} > 10^{14} > 10^{15}$

#synapses [wiki]
Number of Synapses

- **Fruit fly**
- **Honey bee**
- **Mouse**
- **Cat**
- **Macaque**
- **Human**

**NMT with Attention**

Resnet50

[25-50M]

#synapses

[wiki]
Unstoppable Force
Unstoppable Force vs Immovable Object
Scaling Up Neural Networks

**STATISTICAL LEARNING**

Gentlemen, our learner overgeneralizes because the VC-Dimension of our Kernel is too high. Get some experts and minimize the structural risk in a new one. Rework our loss function, make the next kernel stable, unbiased and consider using a soft margin.

**NEURAL NETWORKS**

STACK MORE LAYERS

Machine Learning Memes for Convolutional Teens
Motivation 3:
Neural network scaling and the new understanding of generalization

[Graph showing the relationship between training data set size and generalization error, with three regions: Small Data Region, Power-law Region, and Irreducible Error Region.]
Motivation 3:
Neural network scaling and the new understanding of generalization

Hestness et al. 2018
Motivation 3: Overparameterization and Generalization Theory

Why do large nets generalize better? (predict well on unseen data)

# parameters >> training examples

reducing approximation error ⇒ reduces estimation error
(increase expressivity) (increase generalization)
Motivation 3: 
Overparameterization and Generalization Theory
Deep Double Descent

Neyshabur et al. 2015
Advani and Saxe 2017
Belkin et al. 2018
Motivation 3: Overparameterization and Generalization Theory

Deep Double Descent

Neyshabur et al. 2015
Advani and Saxe 2017
Belkin et al. 2018
Nakkiran et al. 2020

Classical Regime
Bias-Variance Tradeoff

Modern Regime
Larger Model is Better

Critical Regime

Interpolation Threshold

ResNet18 Width Parameter

Test / Train Error

Nakkiran et al. 2020
Motivation 4:
This is a compelling test bed for ML research

Massive multilinguality requires advances in:
- Multi-task learning
- Meta-learning
- Continual learning

To achieve massive multilinguality, we need massive scale, requires advances in:
- Model capacity
- Trainability and optimization
- Efficiency improvements
Agenda

1. Goal & Motivation
2. Project Phases
3. Lessons Learned
To achieve our goal, we knew we had to dramatically scale capacity; but first, we had to enumerate all relevant research challenges.

\[ \text{quality} = f(X, \theta, \mu) \]

- **Data**
  - Any Sequence
  - Arbitrary length

- **Model**
  - Architectures
  - Neural wiring
  - How to parameterize

- **Training**
  - Loss functions
  - Optimizers
  - All the other governing hyper-parameters
To achieve our goal, we knew we had to dramatically scale capacity; but first, we had to enumerate all relevant research challenges.
After our pilot studies, we moved to realistic scenario: M4 in the wild

**RESEARCH PRIORITIES**

- Develop baselines
- Learn, given data imbalance
- Increase model capacity

**GOAL**

Train a 103-language model; attain parity with baselines

Let's go into detail on each priority
We trained and evaluated new bilingual models as controls

1/ Develop baselines:

Data distribution over language pairs

Translation quality of 102 bilingual baselines

High Resource Languages
{Spanish, French, German, ...}

Low Resource Languages
{Yoruba, Sindhi, Hawaiian, ...}
1/ Develop baselines:

**Models**

Wiring: Transformer as explained in [1][2] and [3]

1/ Develop baselines:

Models

Parameter Sharing: Full (rightmost)
2/ Learn, given data imbalance:

Initial baseline on Any→Any model

Any→En translation performance with multilingual baselines

Any to English
Importance of re-balancing data

2/ Learn, given data imbalance:
Increasing the number of languages at the same capacity results in worse quality due to interference, especially in En→Any.
To reduce quality losses, we needed greater model capacity

Relevant research questions:

- Do deep or wide networks drive greater quality gains?
- How deep or wide should we go?
- What quality boost do we attain by sacrificing En→Any (i.e. half of our tasks)?
Massive Neural Networks

Training Deeper Neural Machine Translation Models with Transparent Attention, Bapna et al. EMNLP 2018

GPipe: Efficient Training of Giant Neural Networks using Pipeline Parallelism, Huang et al. NeurIPS 2019

Outrageously Large Neural Networks: The Sparsely-Gated Mixture-of-Experts Layer, Shazeer et al. ICLR 2017

Controlling Computation versus Quality for Neural Sequence Models, Bapna et al. 2020
Number of Synapses

Fruit fly

Honey bee

Mouse

Cat

Macaque

Human

$10^6 > 10^9 > 10^{12} > 10^{13} > 10^{14} > 10^{15}$

#synapses [wiki]
Number of Synapses

NMT with Attention
Resnet50
[25-50M]

#synapses

Fruit fly
Honey bee
Mouse
Cat
Macaque
Human

\[ \text{Number of Synapses} = \begin{cases} \text{Fruit fly} & 10^6 > \\ \text{Honey bee} & 10^9 \\ \text{Mouse} & 10^{12} \\ \text{Cat} & 10^{13} \\ \text{Macaque} & 10^{14} \\ \text{Human} & 10^{15} \end{cases} \]
Number of Synapses

- Fruit fly
- Honey bee
- Mouse
- Cat
- Macaque
- Human

Transformer [400M]
Number of Synapses

- Fruit fly
- Honey bee
- Mouse
- Cat
- Macaque
- Human

Facebook ResNeXt101
Open AI GPT2
MSR ZeRO
NVidia Megatron-LM
Google-T5
[1-10B]

Transformer
[400M]

#synapses [wiki]
Number of Synapses

- **Fruit fly**
- **Honey bee**
- **Mouse**
- **Cat**
- **Macaque**
- **Human**

**Transformer [400M]**

- **Facebook ResNeXt101**
- **Open AI GPT2**
- **MSR ZeRO**
- **NVidia Megatron-LM**
- **Google-T5 [1-10B]**

**M4: Massively Multilingual Massive Machine Translation [80-600B]**

#synapses [wiki]
Do deep or wide networks drive greater quality gains?

Any→En translation performance with model size

3/ Increase model capacity:

Higher quality for deep vs. wide at same capacity
3/ Increase model capacity:

How deep or wide should we go?

All-to-English Translation Quality for M4 Models.

Greatest quality with 128-layer GPipe
Dense Scaling
- GPipe allowed us to break single-TPU memory limit efficiently with pipelining.

Sparse Scaling
- *Conditional computation*, where only part of the network is active for a given example became crucial after certain point.
As we increase number of experts, training and inference scales sublinearly:

- \[ \text{FLOPS} \propto \text{batch}_\text{size} \times \text{avg. gated subnetwork size} \]

Tradeoffs of 512 token-level experts & 1 expert/core:

- Don’t need MoE gradient aggregation
- MoE weight update broadcast is not required
- However, an extra network is required to dispatch activations

3/ Increase model capacity:

How can we scale to 1k chips and beyond?
Conditional computation and Mixture-of-Experts with 50B+ weights
With MoE, bigger is better: $50B \ 512$-MoE > $10B \ 128$-MoE

3/ Increase model capacity:

![Bilingual baselines](image)
At the end of the first phase, two additional insights helped inform research questions for the next phases.

1/ We need a future model that can get the “best of both worlds” across MoE and Deep Transformer.

2/ Scaling up model capacity doesn’t immediately improve performance on En→Any.

[Graphs showing performance metrics across different scenarios and model configurations.]
Final Phase 1 results

3/ Increase model capacity:

Bilingual baselines

<table>
<thead>
<tr>
<th>Model Configuration</th>
<th>BLEU Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bilingual Baselines - 400M parameters</td>
<td></td>
</tr>
<tr>
<td>TransformerMoE(36 Layers, 512 Experts) - 150B (trained on Any-to-En)</td>
<td></td>
</tr>
<tr>
<td>TransformerMoE(12 Layers, 512 Experts) - 50B (trained on Any-to-En)</td>
<td></td>
</tr>
<tr>
<td>TransformerMoE(12 Layers, 128 Experts) - 10B (trained on Any-to-Any)</td>
<td></td>
</tr>
<tr>
<td>Transformer(128 layers, 36k wide, 32 heads) - 6B (trained on Any-to-Any)</td>
<td></td>
</tr>
<tr>
<td>Transformer(12 layers, 8k wide, 16 heads) - 400M (trained on Any-to-Any)</td>
<td></td>
</tr>
</tbody>
</table>
Summary of recent publications

Conference Publications

- NeurIPS
- EMNLP
- AAAI

Massively Multilingual Neural Machine Translation in the Wild: Findings and Challenges

Naveen Arivazhagan * Ankur Bapna * Orhan Firat *
Dmitry Lepikhin Melvin Johnson Maxim Krikun Mia Xu Chen Yuan Cao
George Foster Colin Cherry Wolfgang Macherey Zhifeng Chen Yonghui Wu

GPipe: Easy Scaling with Micro-Batch Pipeline Parallelism

Evaluating the Cross-Lingual Effectiveness of Massively Multilingual Neural Machine Translation

Investigating Multilingual NMT Representations at Scale

Simple, Scalable Adaptation for Neural Machine Translation

Ankur Bapna Naveen Arivazhagan Orhan Firat
Google AI
{ankurbapn,navari,orhanf}@google.com
Drawing the Map of Languages

Investigating Multilingual NMT Representations at Scale, Kudugunta et al. - EMNLP’19
Under the hood ...

When you look inside

a ML model
If we plot M4 language representations, they cluster based on linguistic similarity.
Linguistic similarity determines the representation affinity, especially in higher layers

Representations of Slavic and Turkic languages with Roman and Cyrillic scripts

More details: [Investigating Multilingual NMT Representations at Scale](https://example.com), Kudugunta et al. - EMNLP’19 [SLIDES]
Linguistic similarity determines the representation affinity, especially in higher layers

**Representations of Slavic and Turkic languages with Roman and Cyrillic scripts**

Plot of encoder embedding representations (lowest layer)

Plot of encoder output representations (highest layer)

More details: Investigating Multilingual NMT Representations at Scale, Kudugunta et al. - EMNLP’19 [SLIDES]
Takeaways

1. *Which factors determine the extent of overlap in the learned representations?* Linguistic similarity, not just lexical overlap.

2. *Is the extent of representational overlap similar throughout the model?* It changes, depending on the type of task.

3. *How robust are multilingual NMT representations? (to fine-tuning on another language)*
   Depends on resource size and linguistic similarity to fine-tuning language.
Cross-lingual Downstream Transfer

Evaluating the Cross-Lingual Effectiveness of Massively Multilingual Neural Machine Translation, Siddhant et al. AAAI 2020

XTREME: A Massively Multilingual Multi-task Benchmark for Evaluating Cross-lingual Generalization, Hu et al. ICML 2020
We evaluated M4 representations on downstream tasks; M4 encoder worked better than multilingual BERT in 4 out of 5 tasks.

**Quality: Multilingual BERT vs. M4 encoder**

M4 representations transfer to low-resource languages better than to high-resource languages

**Quality: Multilingual BERT vs. M4 encoder**

Cross-lingual Natural Language Inference (XNLI)

Part of Speech (POS) Tagging

Hypotheses why transfer is better for low-resource languages:
- Translation to English from all languages implicitly forces their representation to be in the same space
- High resource languages are harder to move from point of convergence on fine-tuning
Next, on more tasks and languages (9 tasks, 40 languages)

**XTREME Benchmark:** [sites.research.google/xtreme](http://sites.research.google/xtreme)

---

### Cross-lingual zero-shot transfer (models are trained on English data)

<table>
<thead>
<tr>
<th>Model</th>
<th>Avg</th>
<th>Pair sentence</th>
<th>Structured prediction</th>
<th>Question answering</th>
<th>Sentence retrieval</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>XNLI</td>
<td>PAWS-X</td>
<td>XQuAD</td>
<td>BUCC</td>
</tr>
<tr>
<td></td>
<td></td>
<td>POS</td>
<td>NER</td>
<td>MLQA</td>
<td>Tatoeba</td>
</tr>
<tr>
<td></td>
<td></td>
<td>F1</td>
<td>F1</td>
<td>F1 / EM</td>
<td>F1 / EM</td>
</tr>
<tr>
<td></td>
<td></td>
<td>F1 / EM</td>
<td>F1 / EM</td>
<td>F1 / EM</td>
<td></td>
</tr>
<tr>
<td>mBERT</td>
<td>59.6</td>
<td>65.4</td>
<td>81.9</td>
<td>62.2</td>
<td>59.7 / 43.9</td>
</tr>
<tr>
<td></td>
<td></td>
<td>70.3</td>
<td>64.5 / 49.4</td>
<td>61.4 / 44.2</td>
<td>56.7</td>
</tr>
<tr>
<td></td>
<td></td>
<td>61.4 / 44.2</td>
<td>59.7 / 43.9</td>
<td></td>
<td>38.7</td>
</tr>
<tr>
<td>XLM</td>
<td>55.5</td>
<td>69.1</td>
<td>80.9</td>
<td>61.2</td>
<td>48.5 / 32.6</td>
</tr>
<tr>
<td></td>
<td></td>
<td>70.1</td>
<td>59.8 / 44.3</td>
<td>48.5 / 32.6</td>
<td>56.8</td>
</tr>
<tr>
<td></td>
<td></td>
<td>61.2</td>
<td>59.8 / 44.3</td>
<td></td>
<td>32.6</td>
</tr>
<tr>
<td>XLM-R Large</td>
<td>68.1</td>
<td>79.2</td>
<td>86.4</td>
<td>65.4</td>
<td>65.1 / 45.0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>72.6</td>
<td>76.6 / 60.8</td>
<td>71.6 / 53.2</td>
<td>66.0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>65.4</td>
<td>65.1 / 45.0</td>
<td></td>
<td>57.3</td>
</tr>
<tr>
<td>MMTE</td>
<td>59.3</td>
<td>67.4</td>
<td>81.3</td>
<td>72.3</td>
<td>58.1 / 43.8</td>
</tr>
<tr>
<td></td>
<td></td>
<td>72.3</td>
<td>64.4 / 46.2</td>
<td>60.3 / 41.4</td>
<td>59.8</td>
</tr>
<tr>
<td></td>
<td></td>
<td>58.3</td>
<td>58.1 / 43.8</td>
<td></td>
<td>37.9</td>
</tr>
</tbody>
</table>

### Translate-train (models are trained on English training data translated to the target language)

<table>
<thead>
<tr>
<th>Model</th>
<th>Avg</th>
<th>Pair sentence</th>
<th>Structured prediction</th>
<th>Question answering</th>
<th>Sentence retrieval</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>XNLI</td>
<td>PAWS-X</td>
<td>XQuAD</td>
<td>BUCC</td>
</tr>
<tr>
<td></td>
<td></td>
<td>POS</td>
<td>NER</td>
<td>MLQA</td>
<td>Tatoeba</td>
</tr>
<tr>
<td></td>
<td></td>
<td>F1</td>
<td>F1</td>
<td>F1 / EM</td>
<td>F1 / EM</td>
</tr>
<tr>
<td></td>
<td></td>
<td>F1 / EM</td>
<td>F1 / EM</td>
<td>F1 / EM</td>
<td></td>
</tr>
<tr>
<td>mBERT</td>
<td>-</td>
<td>74.0</td>
<td>86.3</td>
<td>70.0 / 56.0</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>-</td>
<td>65.6 / 48.0</td>
<td>55.1 / 42.1</td>
<td></td>
</tr>
<tr>
<td>mBERT, multi-task</td>
<td>-</td>
<td>75.1</td>
<td>88.9</td>
<td>72.4 / 58.3</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>-</td>
<td>67.6 / 49.8</td>
<td>64.2 / 49.3</td>
<td></td>
</tr>
</tbody>
</table>

### Translate-test (models are trained on English data and evaluated on target language data translated to English)

<table>
<thead>
<tr>
<th>Model</th>
<th>Avg</th>
<th>Pair sentence</th>
<th>Structured prediction</th>
<th>Question answering</th>
<th>Sentence retrieval</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>XNLI</td>
<td>PAWS-X</td>
<td>XQuAD</td>
<td>BUCC</td>
</tr>
<tr>
<td></td>
<td></td>
<td>POS</td>
<td>NER</td>
<td>MLQA</td>
<td>Tatoeba</td>
</tr>
<tr>
<td></td>
<td></td>
<td>F1</td>
<td>F1</td>
<td>F1 / EM</td>
<td>F1 / EM</td>
</tr>
<tr>
<td></td>
<td></td>
<td>F1 / EM</td>
<td>F1 / EM</td>
<td>F1 / EM</td>
<td></td>
</tr>
<tr>
<td>BERT-large</td>
<td>-</td>
<td>76.5</td>
<td>84.4</td>
<td>76.3 / 62.1</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>-</td>
<td>72.9 / 55.3</td>
<td>72.1 / 56.0</td>
<td></td>
</tr>
</tbody>
</table>

### In-language models (models are trained on the target language training data)

<table>
<thead>
<tr>
<th>Model</th>
<th>Avg</th>
<th>Pair sentence</th>
<th>Structured prediction</th>
<th>Question answering</th>
<th>Sentence retrieval</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>XNLI</td>
<td>PAWS-X</td>
<td>XQuAD</td>
<td>BUCC</td>
</tr>
<tr>
<td></td>
<td></td>
<td>POS</td>
<td>NER</td>
<td>MLQA</td>
<td>Tatoeba</td>
</tr>
<tr>
<td></td>
<td></td>
<td>F1</td>
<td>F1</td>
<td>F1 / EM</td>
<td>F1 / EM</td>
</tr>
<tr>
<td></td>
<td></td>
<td>F1 / EM</td>
<td>F1 / EM</td>
<td>F1 / EM</td>
<td></td>
</tr>
<tr>
<td>mBERT, 1000 examples</td>
<td>-</td>
<td>-</td>
<td>87.6</td>
<td>77.9</td>
<td>58.7 / 46.5</td>
</tr>
<tr>
<td>mBERT</td>
<td>-</td>
<td>-</td>
<td>89.8</td>
<td>88.3</td>
<td>74.5 / 62.7</td>
</tr>
<tr>
<td>mBERT, multi-task</td>
<td>-</td>
<td>-</td>
<td>91.5</td>
<td>89.1</td>
<td>77.6 / 68.0</td>
</tr>
<tr>
<td>Human</td>
<td>-</td>
<td>92.8</td>
<td>97.5</td>
<td>91.2 / 82.3</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>-</td>
<td>91.2 / 82.3</td>
<td>90.1 / -</td>
<td></td>
</tr>
</tbody>
</table>

---

On training and hyper-parameters
Adaptive Schedules have the potential to overcome task-scheduling challenges

- **Meta-Learning** (for parallel transfer)
  - Learning to Schedule Tasks
    - Learning Task Weights [paper]
  - Explicit schedules: not scalable
    \[ w_i = \frac{1}{\left( \min \left( 1, \frac{s_i}{b_i} \right)^\alpha + \epsilon \right)} \]
  - Implicit schedules: coupled with optimization
    \[ w_i = 1 + \left( \text{sign} \left( \overline{S} - S_i \right) \right) \min \left( \gamma, \left( \max_j S_j \right)^\alpha \right) |S_i - \overline{S}|^\beta \]

Adaptive Scheduling for Multi-task Learning, Jean et al. NeurIPS CLW 2018
If we meta-learn some hyper-parameters, they don’t require further tuning at scale

- Apply gradient descent on the learning rate (+underlying optimizer)

\[
\frac{\partial f(\theta_{t-1})}{\partial \alpha} = \nabla f(\theta_{t-1}) \cdot \frac{\partial (\theta_{t-2} - \alpha \nabla f(\theta_{t-2}))}{\partial \alpha} = \nabla f(\theta_{t-1}) \cdot (-\nabla f(\theta_{t-2}))
\]

\[
\alpha_t = \alpha_{t-1} - \beta \frac{\partial f(\theta_{t-1})}{\partial \alpha} = \alpha_{t-1} + \beta \nabla f(\theta_{t-1}) \cdot \nabla f(\theta_{t-2})
\]

- Comparison
  - Single pair (wmt’19 en-de): HG ~ Baseline
  - Multi-task (wmt en-{de,fr}): HG  > Baseline
  - BERT: HG ~ Baseline
We also summarized Phase 1 findings in a Google AI blog post ([link](https://ai.googleblog.com/2019/10/exploring-massively-multilingual-massive-neural-machine-translation.html)).

---

**Exploring Massively Multilingual, Massive Neural Machine Translation**

Friday, October 11, 2019

Posted by Ankur Bapna, Software Engineer and Orhan Firat, Research Scientist, Google Research

"... perhaps the way of translation is to descend, from each language, down to the common base of human communication — the real but as yet undiscovered universal language — and then re-emerge by whatever particular route is convenient.” — Warren Weaver, 1949
Challenges & Open Problems

Cross-lingual Downstream Transfer Learning
- Better learning objectives
- New tasks and languages

Unsupervised Machine Translation
- M4 as a knowledge base for unsupervised MT
- Adapting to unseen languages (modalities)

Transfer Learning
- Parallel transfer: smarter schedulers, mitigating interference
- Serial transfer: continual/meta learning, maximize transfer w/o forgetting
Next 1000 Languages

Towards universal translation
Multilingual models disproportionately boost low-resource quality, getting close to so-called “interlingua”

There’s also been tremendous progress on unsupervised MT, but deficiencies remain (e.g., domain mismatch, distant languages)

We need the confluence of multilingual and unsupervised MT research to build a universal translation model

But first, we need the right data, or just some data!
2/ Solve supervision challenges

As we move beyond 100 languages, we journey over supervised, semi-supervised and unsupervised learning.

![Graph showing the number of examples for different languages in parallel and monolingual settings.](image)
2/ Solve supervision challenges
As we move beyond 100 languages, we journey over supervised, semi-supervised and unsupervised learning.

😊: Symbiosis between supervised and unsupervised MT
**Solution:** Supervised + Self-supervised loss simultaneously, boosting quality even without iterative back-translation

Combining **Supervised** and **Self-Supervised** Learning, enables M4 model to learn new unseen languages with **monolingual data only**.

- **Lithuanian → English**
  - Multilingual Model with Latvian
- **Latvian → English**
  - **Multilingual Model W/O Latvian + monolingual Latvian data**
- **Hindi → English**
Adding Monolingual data also enhances Zero-shot MT on Non-English Centric Language Pairs too
Bonus Dataset: M4-Public

About to release a benchmark on:

- Compilation of public datasets - **125 languages**
  - Europarl, Paracrawl, Ted57, JW300, Newscrawl, ...
  - Diverse set of languages, coverage
  - Multi-domain, noisy, imbalanced
  - 450M parallel data

- A new evaluation set - **60 languages**
  - Professional translations
  - Multi-way parallel
  - Web-domain

- Enables studying
  - Representation analysis
  - Out-of-domain generalization
  - Learning dynamics of massively multi-task models
  - Supervised, semi-supervised and unsupervised MT
Agenda

1. Motivation

2. Project Phases

3. Lessons Learned
Workflow
Workflow of AI Researchers working at Scale

- Models: Expressivity, Robustness, Modularity
- Data: Data Selection, Collection, Filtering
- Scale: Increased Capacity, Tools, Infra
- Trainability: Optimization, Stabilization, Understanding
1/ Lesson Learned

Tools and Systems in order to orchestrate a giant machinery is essential for large scale ML projects
Components of a ML System

Training
- Reads the data, computes loss and gradients, applies parameter update.
- The most compute intensive job, runs on TPU

Inference
- Reads a checkpoint of the trained model, runs inference (beam-search)
- Generating output sequences, usually runs on GPU

Evaluation
- Reads a checkpoint of the trained model, computes loss on dev set.
- Used for monitoring the progress, usually runs on GPU or CPU
Under the hood

*This is just a sketch, exact locations are inaccurate.
Under the hood

*This is just a sketch, exact locations are inaccurate.*
Under the hood

*This is just a sketch, exact locations are inaccurate.
Under the hood

*This is just a sketch, exact locations are inaccurate.
Under the hood

*This is just a sketch, exact locations are inaccurate.*
*This is just a sketch, exact locations are inaccurate.
Under the hood

*This is just a sketch, exact locations are inaccurate.
Under the hood

*This is just a sketch, exact locations are inaccurate.*
Tensorflow Lingvo: github.com/tensorflow/lingvo
Prototyping and debugging large scale neural networks may require **travelling off the beaten path**: devising new plots to monitor.
Debugging Large Scale Models

Sources of “bugs” in large scale machine learning:

1. **Regular bugs**: introduced by ML practitioner
   a. Soln. go grab a coffee

2. **Device/Infra bugs**: hideous bugs
   a. Soln. change device, data, data center

3. **Theoretical bugs**: well... this should’ve never happened in the first place
   a. Soln. brush up your ML.
   b. Look at the right thing, norm of the gradient vs norm of the weights.
   c. Isolate initialization, optimization and malicious data.
Hyper-parameter search may no longer be an option if a single model is taking weeks to train on thousands of TPUs: meta-learning.
Hyper-parameter Search

First: No one has infinite resources → the more compute we get, the larger we scale.

Some rule-of-thumbs
- All variables are interconnected: if you are changing one, expect the others to be changed
- Always start with the learning rate, then the batch-size
- Hill-climbing is as good as random search

Some tools to automate
- Vizier for Cloud
- Tune for Pytorch
The Learning Rate Schedules

“Often the single most important hyper-parameter”
Practical recommendations for gradient-based training of deep architectures,
Bengio 2012

Should always be tuned.
Importance of Configs

For large scale experiments:

- Reproducibility is more important than code reuse, cosmetics and other conventions
- Maintaining sufficient checkpoints
- Having experimental results attached to the configs

DO NOT INHERIT CONFIG CLASSES

```python
#model_registry.RegisterSingleTaskModel
class WmtEnDeTransformerBase(base_model_params.SingleTaskModelParams):
    r"""Params for WMT'14 En->De."""

    DATADIR = '/usr/local/google/wmt14/wgm/'
    VOCAB_SIZE = 32000

    @classmethod
def Train(cls):
        p = input_generator.NmtInput.Params()
        p.file_pattern = 'tfrecords':os.path.join(cls.DATADIR, 'train.tfrecords-*)
        p.tokenizer.token_vocab_filepath = os.path.join(cls.DATADIR, 'wgm-en.de.voc')
        p.bucket_batch_limit = [(128, 102, 85, 73, 64, 51, 42)]
        return p

    @classmethod
def Dev(cls):
        p = input_generator.NmtInput.Params()
        p.file_pattern = 'tfrecords':os.path.join(cls.DATADIR, 'dev.tfrecords')
        p.tokenizer.token_vocab_filepath = os.path.join(cls.DATADIR, 'wgm-en.de.voc')
        return p

    @classmethod
def Test(cls):
        p = input_generator.NmtInput.Params()
        p.file_pattern = 'tfrecords':os.path.join(cls.DATADIR, 'test.tfrecords')
        p.tokenizer.token_vocab_filepath = os.path.join(cls.DATADIR, 'wgm-en.de.voc')
        return p

    @classmethod
def Task(cls):
        p = base_config.SetupTransformerParams(
            model.TransformerModel.Params(),
            name='wmt14_en_de_transformer_base',
            vocab_size=cls.VOCAB_SIZE,
            model_dim=512,
            hidden_dim=2048,
            num_heads=8,
            num_layers=6,
            residual_dropout_prob=0.1,
            input_dropout_prob=0.1,
            learning_rate=3.0,
            warmup_steps=40000)
        p.eval.samples_per_summary = 7500
        return p

https://github.com/tensorflow/lingvo/blob/master/lingvo/tasks/mt/params/wmt14_en_de.py
Bonus

Platform independent frameworks
● TF → CPU/GPU/TPU/Mobile/Browser, use the right tool for the job

Don’t get lost in the nuances,
● Ask yourself, which research question I’m trying to answer all the time
● There is no end in optimization, so don’t over-optimize things

Working with larger teams
● Async approaches to sync, a simple spreadsheet can help a lot

We need something post-silicone, given the current consumption of chips!
THANKS!

orhanf@google.com
Appendix
More on Scaling

Hardware, GPipe and Large Batches
GPipe: Easy Scaling with Pipeline Parallelism - Huang et al., 2019

- **Image-Net**
  - Top-1 Accuracy vs. Number of Parameters (Millions)
  - Benchmark architectures: GoogleNet, Inception3, ResNet-152, ResNetXt-101, Senet, AmoebaNetC(6, 228), AmoebaNetB(18, 512)

- **Machine Translation**
  - Average BLEU vs. Number of Parameters (Billions)
  - Benchmark architectures: T(6, 8192, 16), T(12, 16384, 32), T(24, 8192, 16), T(64, 16384, 32)
Compute and Machine Learning Systems

- Training on 1024 TPU-v3 chips
- Bfloat16 (Brain Floating Point)
- GPipe: Micro-Batch Pipeline Parallelism (Huang et al., 2019)
  - Rematerialization (gradient checkpointing)
  - Large batches (4M examples)

TPU v3 Pod in its habitat
Compute - Tensor Processing Units

Cloud TPU v2
- 180 teraflops
- 64 GB High Bandwidth Memory (HBM)

Cloud TPU v3
- 420 teraflops
- 128 GB HBM

Cloud TPU v2 Pod (beta)
- 11.5 petaflops
- 4 TB HBM
- 2-D toroidal mesh network

Cloud TPU v3 Pod (beta)
- 100+ petaflops
- 32 TB HBM
- 2-D toroidal mesh network

https://cloud.google.com/tpu/
Only one accelerator is active when the model is distributed across the accelerators.

F₀ waits for outputs of F₁.

F₁ waits for outputs of F₀.
Performance

Sublinear scaling with or without high speed interconnect.

<table>
<thead>
<tr>
<th></th>
<th>TPU</th>
<th>AmoebaNet</th>
<th>Transformer</th>
</tr>
</thead>
<tbody>
<tr>
<td>$K = 2$</td>
<td>1</td>
<td>1.13</td>
<td>1.38</td>
</tr>
<tr>
<td>$M = 1$</td>
<td>1</td>
<td>1.07</td>
<td>1.3</td>
</tr>
<tr>
<td>$M = 4$</td>
<td>1.07</td>
<td>1.26</td>
<td>1.72</td>
</tr>
<tr>
<td>$M = 8$</td>
<td>1.7</td>
<td>3.2</td>
<td>4.8</td>
</tr>
<tr>
<td>$M = 32$</td>
<td>1.21</td>
<td>1.84</td>
<td>3.48</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1.8</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3.4</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>6.3</td>
</tr>
</tbody>
</table>

K: # of model partitions.
M: # of batch splitting.
Validating recent neural-net theory at scale.

- Implicit acceleration by overparameterization.
- Deeper models are sample efficient.
- Effect of very large batch-sizes.
- Still noise in the 1st order approximation.

<table>
<thead>
<tr>
<th>Batch Size</th>
<th>260K</th>
<th>1M</th>
<th>4M</th>
</tr>
</thead>
<tbody>
<tr>
<td>BLEU</td>
<td>30.92</td>
<td>31.86</td>
<td>32.71</td>
</tr>
<tr>
<td>Loss (NLL)</td>
<td>2.58</td>
<td>2.51</td>
<td>2.46</td>
</tr>
</tbody>
</table>

- Way to go for large datasets,
  - Risk of injecting too many noisy examples
Making M4 Practical: Simple Adaptation

Simple, Scalable Adaptation for Neural Machine Translation, Bapna et al. - EMNLP’19
Residual Adapters allow for improving quality and for specializing models on languages or domains.

Modules that specialize model for these languages

Residual Adapters: overview
With Residual Adapters, we regain the quality drop in high-to-mid resource languages and enable domain-adaptation.

Relevant paper: [Simple, Scalable Adaptation for Neural Machine Translation](https://emnlp'19) (Bapna et al.)
Restore quality on high-to-mid resource languages with 13% extra parameters per language

Relevant paper: Simple, Scalable Adaptation for Neural Machine Translation (Bapna et al. - EMNLP’19)
For high resource languages we see further gains by increasing adapter capacity, although the returns are diminishing.

Relevant paper: Simple, Scalable Adaptation for Neural Machine Translation (Bapna et al. - EMNLP’19)
Enhancing Zero-Shot Translation: The Missing Ingredient

Enhancing Zero-Shot Translation: The Missing Ingredient

Bridging the gap between pivoting and zero-shot
- auxiliary losses on the NMT encoder
- impose representational invariance via loss
- easy scalability to multiple languages

<table>
<thead>
<tr>
<th></th>
<th>vanilla</th>
<th>align(cosine)</th>
</tr>
</thead>
<tbody>
<tr>
<td>en ↔ xx (8)</td>
<td>30.11</td>
<td>- 29.95</td>
</tr>
<tr>
<td>xx ↔ yy (12)</td>
<td>16.73 (zs)</td>
<td>17.76</td>
</tr>
<tr>
<td>All (20)</td>
<td>22.2</td>
<td>22.81</td>
</tr>
</tbody>
</table>

Table 5: Average BLEU scores for multilingual model on IWSLT-2017; Zero-Shot results are marked (zs).