A call to build models like we build open-source software

Colin Raffel
Unsupervised pre-training

Supervised fine-tuning

minivan  pizza  soap  terrapin

morel  blewit  puffball  death cap
The cabs charged the same rates as those used by horse-drawn cabs and were initially quite popular; even the Prince of Wales (the future King Edward VII) travelled in one. The cabs quickly became known as "hummingbirds" for the noise made by their motors and their distinctive black and yellow livery.

This movie is terrible! The acting is bad and I was bored the entire time. There was no plot and nothing interesting happened. I was really surprised since I had very high expectations. I want 103 minutes of my life back!
SQuAD Exact Match score

from https://paperswithcode.com/sota/question-answering-on-squad11-dev
Transfer learning vs. No transfer learning for Exact Match Score.

From https://paperswithcode.com/sota/question-answering-on-squad11-dev
TIMIT Phoneme Error Rate

No transfer learning

Transfer learning

from https://paperswithcode.com/sota/speech-recognition-on-timit
We provide pre-trained models, using the PyTorch `torch.utils.model_zoo`. These can be constructed by passing `pretrained=True`:

```python
import torchvision.models as models
resnet18 = models.resnet18(pretrained=True)
alexnet = models.alexnet(pretrained=True)
squeezenet = models.squeezenet1_0(pretrained=True)
vgg16 = models.vgg16(pretrained=True)
densenet = models.densenet161(pretrained=True)
inception = models.inception_v3(pretrained=True)
google_net = models.googlenet(pretrained=True)
shufflenet = models.shufflenet_v2_x1_0(pretrained=True)
```
GPT-3 175B model required 3.14E23 FLOPS of computing for training. Even at theoretical 28 TFLOPS for V100 and lowest 3 year reserved cloud pricing we could find, this will take 355 GPU-years and cost $4.6M for a single training run.

from https://lambdalabs.com/blog/demystifying-gpt-3/
<table>
<thead>
<tr>
<th>Model</th>
<th>Upload Date</th>
<th>Downloads</th>
<th>Favorites</th>
</tr>
</thead>
<tbody>
<tr>
<td>bert-base-uncased</td>
<td>May 18</td>
<td>30M</td>
<td>54</td>
</tr>
<tr>
<td>roberta-large</td>
<td>May 21</td>
<td>13.1M</td>
<td>20</td>
</tr>
<tr>
<td>distilbert-base-uncased</td>
<td>Aug 29</td>
<td>4.83M</td>
<td>26</td>
</tr>
<tr>
<td>xlm-roberta-base</td>
<td>Sep 16</td>
<td>4.78M</td>
<td>11</td>
</tr>
<tr>
<td>bert-base-cased</td>
<td>Sep 6</td>
<td>4.02M</td>
<td>6</td>
</tr>
<tr>
<td>distilbert-base-uncased-finetuned-sst-2-english</td>
<td>Feb 9</td>
<td>3.54M</td>
<td>18</td>
</tr>
<tr>
<td>roberta-base</td>
<td>Jul 6</td>
<td>3.45M</td>
<td>6</td>
</tr>
<tr>
<td>gpt2</td>
<td>May 19</td>
<td>3.34M</td>
<td>24</td>
</tr>
</tbody>
</table>
OpenAI technology, just an HTTPS call away

Apply our API to any language task — semantic search, summarization, sentiment analysis, content generation, translation, and more — with only a few examples or by specifying your task in English.
UnifiedQA

Additional training
T5

UnifiedQA

MACAW

Additional training
mT5

ByT5

UnifiedQA

UNICORN

T5

T5+LM

mT5

ByT5

UnifiedQA

UNICORN

T5+LM

T0

--- Additional training

--- New model
Fork
from https://dvc.org/
How can we enable collaborative and continual development of machine learning models?

We need to be able to cheaply communicate patches and merge updates from different contributors.
How can we enable collaborative and continual development of machine learning models?

We need to be able to cheaply communicate patches and merge updates from different contributors.
\[ D_{KL}(p_\theta(y|x) \mid\mid p_{\theta+\delta}(y|x)) \]
\[ D_{KL}(p_{\theta}(y|x) \mid\mid p_{\theta+\delta}(y|x)) \]

\[ \mathbb{E}_x D_{KL}(p_{\theta}(y|x) \mid\mid p_{\theta+\delta}(y|x)) = \delta^T F_{\theta} \delta + O(\delta^3) \]
\[ D_{KL}(p_{\theta}(y|x) \parallel p_{\theta+\delta}(y|x)) \]

\[ \mathbb{E}_x D_{KL}(p_{\theta}(y|x) \parallel p_{\theta+\delta}(y|x)) = \delta^T F_{\theta} \delta + O(\delta^3) \]

\[ F_{\theta} = \mathbb{E}_{x \sim p(x)} \left[ \mathbb{E}_{y \sim p_{\theta}(y|x)} \nabla_\theta \log p_{\theta}(y|x) \nabla_\theta \log p_{\theta}(y|x)^T \right] \]
\[ D_{KL}(p_\theta(y|x) \parallel p_{\theta+\delta}(y|x)) \]

\[ \mathbb{E}_x D_{KL}(p_\theta(y|x) \parallel p_{\theta+\delta}(y|x)) = \delta^T F_\theta \delta + O(\delta^3) \]

\[ F_\theta = \mathbb{E}_{x \sim p(x)} \left[ \mathbb{E}_{y \sim p_\theta(y|x)} \nabla_\theta \log p_\theta(y|x) \nabla_\theta \log p_\theta(y|x)^T \right] \]

\[ \hat{F}_\theta = \frac{1}{N} \sum_{i=1}^{N} \mathbb{E}_{y \sim p_\theta(y|x_i)} \left( \nabla_\theta \log p_\theta(y|x_i) \right)^2 \]
Fisher-Induced Sparse Unchanging (FISH) Mask

$$\mathbb{E}_x \mathbb{E}_y (\nabla_{\theta} p_\theta(y|x))^2$$

Top-k
<table>
<thead>
<tr>
<th>Method</th>
<th>Sparsity</th>
<th>GLUE Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dense Fine-tuning</td>
<td>100%</td>
<td>82.5</td>
</tr>
<tr>
<td>Bit-Fit</td>
<td>0.08%</td>
<td>81.2</td>
</tr>
<tr>
<td>FISH Mask</td>
<td>0.08%</td>
<td>81.3</td>
</tr>
<tr>
<td>Diff Pruning</td>
<td>0.50%</td>
<td>81.5</td>
</tr>
<tr>
<td>FISH Mask</td>
<td>0.50%</td>
<td>82.6</td>
</tr>
</tbody>
</table>
How can we enable collaborative and continual development of machine learning models?

We need to be able to cheaply communicate patches and merge updates from different contributors.
Pre-training → Downstream
Pre-training → Intermediate → Downstream

Pre-training → Intermediate → Downstream
\[
\arg\max_\theta \sum_{i=1}^{M} \lambda_i \log p(\theta | D_i)
\]
$$\arg \max_{\theta} \sum_{i=1}^{M} \lambda_i \log p(\theta | D_i)$$

Log posterior for model \(i\)
\[
\arg \max_{\theta} \sum_{i=1}^{M} \lambda_i \log p(\theta | D_i)
\]

Hyperparameter controlling the importance of model \(i\)
\[ \arg \max_\theta \sum_{i=1}^{M} \lambda_i \log p(x_i) \]
\[
\arg \max_{\theta} \sum_{i=1}^{M} \lambda_i \log \mathcal{N}(\theta | \theta_i, H_i^{-1})
\]
$$\arg\max_\theta \sum_{i=1}^{M} \lambda_i \log \mathcal{N}(\theta | \theta_i, 1)$$
$$\arg\max_{\theta} \sum_{i=1}^{M} \lambda_i \log \mathcal{N}(\theta | \theta_i, F_i)$$
\[
\arg\max_\theta \sum_{i=1}^{M} \lambda_i \log \mathcal{N}(\theta | \theta_i, \Sigma_i)
\]
\[ \arg\max_{\theta} \sum_{i=1}^{M} \lambda_i \log \mathcal{N}(\theta|\theta_i, \hat{F}_i) \]
\[
\arg\max_{\theta} \sum_{i=1}^{M} \lambda_i \log \mathcal{N}(\theta|\theta_i, \hat{F}_i)
\]
Intermediate Target

![Diagram showing relationships between intermediate and target tasks with heat map indicating task similarities.]
<table>
<thead>
<tr>
<th>Task</th>
<th>Unmerged</th>
<th>Merged</th>
<th>Fine-tuned</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHEMPROT</td>
<td>82.7_{0.3}</td>
<td>83.1_{0.4}</td>
<td>82.5_{0.1}</td>
</tr>
<tr>
<td>ACL-ARC</td>
<td>70.5_{3.2}</td>
<td>73.2_{1.7}</td>
<td>71.5_{3.0}</td>
</tr>
<tr>
<td>SciERC</td>
<td>81.0_{0.4}</td>
<td>81.3_{0.5}</td>
<td>81.6_{1.0}</td>
</tr>
</tbody>
</table>
How can we enable collaborative and continual development of machine learning models?

We need to be able to cheaply communicate patches and merge updates from different contributors.
How can we enable collaborative and continual development of machine learning models?

We need to be able to rapidly evaluate proposed changes to the model to ensure backward compatibility.
How can we enable collaborative and continual development of machine learning models?

We need to be able to combine modular components of different models to provide new skills and capabilities.
Training Neural Networks with Fixed Sparse Masks
Yi-Lin Sung, Varun Nair, and Colin Raffel

Merging Models with Fisher-Weighted Averaging
Michael Matena and Colin Raffel

Please give me feedback: