

EMERGING TRENDS

# Emerging trends: A gentle introduction to fine-tuning

Kenneth Ward Church<sup>1\*</sup>, Zeyu Chen<sup>2</sup> and Yanjun Ma<sup>2</sup>

<sup>1</sup>Baidu, Sunnyvale, CA, USA and <sup>2</sup>Baidu, Beijing, China

\*Corresponding author. E-mail: [kenneth.ward.church@gmail.com](mailto:kenneth.ward.church@gmail.com)

(Received 2 October 2021)

## Abstract

The previous Emerging Trends article (Church *et al.*, 2021. *Natural Language Engineering* 27(5), 631–645.) introduced deep nets to poets. Poets is an imperfect metaphor, intended as a gesture toward inclusion. The future for deep nets will benefit by reaching out to a broad audience of potential users, including people with little or no programming skills, and little interest in training models. That paper focused on inference, the use of pre-trained models, as is, without fine-tuning. The goal of this paper is to make fine-tuning more accessible to a broader audience. Since fine-tuning is more challenging than inference, the examples in this paper will require modest programming skills, as well as access to a GPU. Fine-tuning starts with a general purpose base (foundation) model and uses a small training set of labeled data to produce a model for a specific downstream application. There are many examples of fine-tuning in natural language processing (question answering (SQuAD) and GLUE benchmark), as well as vision and speech.

**Keywords:** Deep nets; Pre-training; Fine-tuning; Benchmarks; GLUE; SQuAD; ImageNet; BERT; ERNIE; wav2vec

## 1. Introduction

This paper will show how to use fine-tuning on a few benchmarks such as SQuAD and GLUE, as well as a task based on ImageNet. In our previous Emerging Trends article on inference (Church *et al.* 2021), we posted code on GitHub<sup>1</sup> because code in blogs and hubs tends to be too demanding for the target audience (poets). This paper will discuss code on PaddleHub<sup>2</sup> and HuggingFaceHub,<sup>3</sup> since these examples are very well done, and the target audience for this paper is more advanced (and less inclusive).

Finally, we will end on a cautionary note. Based on the success of fine-tuning on a number of benchmarks, one might come away with the impression that fine-tuning is all we need. However, we believe the glass is half-full: while there is much that can be done with fine-tuning, there is always more to do. Fine-tuning has become popular recently, largely because it works so well on so many of our most popular benchmarks. But the success of fine-tuning can also be interpreted as a criticism of these benchmarks. Many of these benchmarks tend to focus too much on tasks that are ideal for fine-tuning, and not enough on opportunities for improvement.

<sup>1</sup>[https://github.com/kwchurch/deepnet\\_examples](https://github.com/kwchurch/deepnet_examples).

<sup>2</sup><https://github.com/PaddlePaddle/PaddleHub/tree/release/v2.1/demo>.

<sup>3</sup><https://github.com/huggingface/transformers/tree/master/examples/pytorch>.

**Table 1.** Base models are large in two respects: model size and training data

Pre-trained (foundation) model	Parameters	Training data
ResNet-50 (He <i>et al.</i> 2016)	23M	14M images from ImageNet
VT (Wu <i>et al.</i> 2020)	11.7–21.9M	14M images from ImageNet
Wav2vec (Baevski <i>et al.</i> 2020)	95–317M	960 hours from LibriSpeech
BERT (Devlin <i>et al.</i> 2019)	110–340M	3.3B words from Books/Wiki
ERNIE 2.0 (Sun <i>et al.</i> 2020)	110–340M	7.9B en + 15B zh tokens
ERNIE 3.0 (Sun <i>et al.</i> 2021)	10B	375B tokens of text, as well as knowledge graph
RoBERTa (Liu <i>et al.</i> 2019)	110M	160GBs of text
GPT-2 (Radford <i>et al.</i> 2019)	1.5B	40GBs of text
GPT-3 (Brown <i>et al.</i> 2020)	125M–175B	1TB from Common Crawl, Books and Wikipedia
XLM-RoBERTa (Conneau <i>et al.</i> 2020)	12–16B	2.5TBs from many languages

### 1.1 Pre-training, fine-tuning, and inference

Much of the recent literature in deep nets involves three processes:

1. Pre-training base (foundation) models
  - Language: BERT (Devlin *et al.* 2019), ERNIE (Sun *et al.* 2020; Sun *et al.* 2021)
  - Speech: wav2vec<sup>4</sup> (Baevski *et al.* 2020)
  - Vision: ResNet (He *et al.* 2016), VIT (Wu *et al.* 2020)
2. Fine-tuning (this paper)
3. Inference (Church *et al.* 2021).

There will be relatively little discussion of inference in this paper since that topic was covered in our previous article, except to point out that inference is the least demanding of the three tasks, in terms of both programming skills and computational resources. Inference takes a fine-tuned model as input, as well as features,  $x$ , for a novel input, and outputs a predicted label,  $\hat{y}$ . The examples discussed in footnote 1 are short (10–100 lines of code) and easy to read. They cover a wide range of use cases in natural language processing (sentiment analysis, named entity recognition, question answering (QA/SQuAD), machine translation), as well as use cases in vision and speech. From a commercial perspective, inference is probably more profitable than training.

The literature, however, focuses on training, because training is more challenging than inference. The terminology is still in a state of flux. Base models are sometimes referred to as pre-trained models (PTMs) (Han *et al.* 2021) or Foundation Models (Bommasani *et al.* 2021). These models are typically pre-trained on large quantities of data, often without annotations/labels, as shown in Tables 1–2.

With the exception of GPT-3 and ERNIE 3.0, most PTMs in the literature can be downloaded from hubs such as PaddleHub,<sup>5</sup> HuggingFaceHub,<sup>6</sup> and Fairseq<sup>7</sup> (Ott *et al.* 2019).

<sup>4</sup><https://huggingface.co/blog/fine-tune-wav2vec2-english>.

<sup>5</sup><https://www.paddlepaddle.org.cn/hublist>.

<sup>6</sup><https://huggingface.co/transformers>.

<sup>7</sup><https://github.com/pytorch/fairseq>.

**Table 2.** Some popular datasets for training base models

Dataset	Description
Billion Word Benchmark (Chelba <i>et al.</i> 2013)	A billion words of English
Common Crawl (Buck, Heafield, and van Ooyen 2014)	<a href="https://github.com/commoncrawl">https://github.com/commoncrawl</a>
Book Corpus (Zhu <i>et al.</i> 2015)	Speech with text
ImageNet (Deng <i>et al.</i> 2009)	14M images, annotated with 21k classes
LibriSpeech (Panayotov <i>et al.</i> 2015)	960 hours of speech with text
LJ Speech	<a href="https://keithito.com/LJ-Speech-Dataset/">https://keithito.com/LJ-Speech-Dataset/</a>
AISHELL-2 (Du <i>et al.</i> 2018)	<a href="https://protect-eu.mimecast.com/s/_t4dC0Vqqsjz5YGswPwKH?domain=aishelltech.com">https://protect-eu.mimecast.com/s/_t4dC0Vqqsjz5YGswPwKH?domain=aishelltech.com</a>

Historically, many of these hubs started with models for natural language processing, though they are beginning to expand into other fields such as speech (SpeechBrain<sup>8</sup> (Ravaneli *et al.* 2021) and ESPnet<sup>9</sup> (Watanabe *et al.* 2018; Hayashi *et al.* 2020; Inaguma *et al.* 2020; Li *et al.* 2021)) and vision (ViT<sup>10</sup> (Dosovitskiy *et al.* 2021; Tolstikhin *et al.* 2021; Steiner *et al.* 2021; Chen, Hsieh, and Gong 2021)).

Base models tend to be very general and can be applied to a broad set of downstream applications, which is necessary to make the business case work, since pre-training is expensive.

The terms, *pre-training* and *fine-tuning*, became popular following (Devlin *et al.* 2019), though these terms can be found in earlier work (Howard and Ruder 2018):<sup>11</sup>

*There are two steps in our framework: pre-training and fine-tuning. During pre-training, the model is trained on unlabeled data over different pre-training tasks. For fine-tuning, the BERT model is first initialized with the pre-trained parameters, and all of the parameters are fine-tuned using labeled data from the downstream tasks. Each downstream task has separate fine-tuned models, even though they are initialized with the same pre-trained parameters.* (Devlin *et al.* 2019)

Devlin *et al.* (2019) follow this discussion of pre-training and fine-tuning by introducing a running example in their Figure 1, where a large BERT base model is pre-trained on a large corpus of unlabeled training data. Then, at fine-tuning time, the PTM is refined to produce three separate models for three separate downstream tasks: (a) a question answering task (SQuAD), (b) a named entity task, and (c) a task from the GLUE benchmark, MNLI, that involves textual entailment (Dagan, Glickman, and Magnini 2005). This paper will focus on the fine-tuning proposal in Devlin *et al.* (2019) and will not discuss recently proposed alternatives such as Pattern-Exploiting Training (Schick and Schütze 2021).

The terms, *fine-tuning*, *transfer learning*, and *classification/regression heads*, are similar to one another, though there are some important differences. Fine-tuning is a relatively new term compared to transfer learning.<sup>12</sup> Venues such as NeurIPS (formerly NIPS) used to be more theoretical a decade ago. Older papers such as Pan and Yang (2009) used to provide more definitions; more recent treatments characterize newer terms such as fine-tuning by example without formal definitions.

<sup>8</sup><https://speechbrain.github.io/>.

<sup>9</sup><https://github.com/espnet/espnet>.

<sup>10</sup>[https://github.com/google-research/vision\\_transformer](https://github.com/google-research/vision_transformer).

<sup>11</sup><https://ruder.io/recent-advances-lm-fine-tuning/>.

<sup>12</sup><https://ruder.io/transfer-learning/>.

In contrast to classification/regression heads, fine-tuning implies updates to many/most layers within the model. Classification and regression heads treat the last layer of the model as input features,  $X$ , and predict outputs,  $\hat{Y}$ , using classification (if gold labels,  $Y$ , are symbolic) or regression (if gold labels,  $Y$ , are continuous).

### 1.2 Labeled/unlabeled training data

Fine-tuning is exciting for a number of reasons. One of many reasons is the combination of labeled and unlabeled training data. There has been a long tradition in artificial intelligence (AI), dating back to semi-supervised learning (Zhu 2005) and co-training (Blum and Mitchell 1998) and probably earlier discussions dating back to 1970s, in search of ways to combine small amounts of labeled training data with vast quantities of unlabeled training data. In many use cases, it is possible to find vast quantities of unlabeled data. Since labeled data tend to be much harder to come by, it is desirable to find ways to achieve high performance with as little labeled data as possible.

Labeled data tend to be expensive in a number of respects: money, time, grief, unreliable labels, and unfair labor practices. It has become standard practice to use Amazon Mechanical Turk to label data (Callison-Burch 2009; Finin *et al.* 2010; Pavlick *et al.* 2014; Nangia and Bowman 2019), though there are many concerns with this practice, including questions about the so-called “gig economy” and invisible labor (Hara *et al.* 2018).<sup>13,14</sup>

In language applications, pre-training typically uses unlabeled data and fine-tuning uses labeled data. In vision and speech, both pre-training and fine-tuning are based on labeled data, though the labels for pre-training may be quite different from the labels for fine-tuning. In Section 2.1, for example, we will discuss a vision classification task where pre-training uses 1000 classes from ImageNet and fine-tuning uses 5 different classes (flowers). It is amazing that fine-tuning works as well as it does, even when fine-tuning is given a surprisingly small training set of flowers. It appears that fine-tuning is “unreasonably effective” in transferring knowledge from pre-trained base models (ResNet for ImageNet) to novel tasks (flower classification).

It is common to refer to training with labels as *supervised* learning and training without labels as *unsupervised* learning. Supervised methods are provided both input features,  $x$ , and output labels,  $y$ , at training time, whereas unsupervised methods are given  $x$  but not  $y$ . In both cases, the task is to learn a model that can be applied to novel inputs to predict labels,  $\hat{y}$ , at inference time. Evaluations compare the predicted labels,  $\hat{y}$ , to gold labels,  $y$ , in a held out test set.

### 1.3 Balance, samples, and populations of interest

Fine-tuning offers a promising way to deal with various realities. Domain shift happens. It is very likely that a real user will ask a real product to do something that is very different from the training data. It is also likely that PTMs will need to be updated periodically, as the world changes.

Of course, it would be better to pre-train on more relevant data in the first place. In practice, the data for pre-training tend to be more catch-as-catch-can than a balanced corpus. Balance is taken more seriously in lexicography than machine learning:

*It was also believed that quality (balance) mattered, but there were few, if any, empirical studies to justify such beliefs. It was extremely controversial when engineers such as Mercer questioned these deeply held beliefs in 1985<sup>15</sup> with: “there is no data like more data.” Most people working on corpus-based methods in lexicography were deeply committed to balance as a matter of faith, and were deeply troubled by Mercer’s heresy. (Church and Bian 2021)*

<sup>13</sup>[https://nips.cc/virtual/2020/public/invited\\_16164.html](https://nips.cc/virtual/2020/public/invited_16164.html).

<sup>14</sup><https://www.youtube.com/watch?v=o5zg57nKqGE>.

<sup>15</sup><http://www.lrec-conf.org/lrec2004/doc/jelinek.pdf>.

In contrast to datasets such as those in Table 2, lexicographers prefer balanced corpora such as the Brown Corpus (Kučera and Francis 1967; Francis and Kučera 1979; Francis and Kučera 1982) and British National Corpus<sup>16</sup> (BNC) (Aston and Burnard 1998; Burnard 2002). The Brown Corpus was designed to be a sample of contemporary American English. The BNC is similar but for British English. The BNC is also larger and more recent, though neither the Brown Corpus nor BNC would be considered large or recent by modern standards. The Brown Corpus contains 1M words from the 1960s, and the BNC contains 100M words from the 1990s.

There is some evidence from psycholinguistics supporting both sides of the debate between balance and catch-as-catch-can; correlations of corpus statistics and psycholinguistic norms tend to improve with quality (balance) as well as quantity (corpus size) (Rapp 2014a; Rapp 2014b).

Fine-tuning could be viewed as a mechanism to correct for mismatches between training data and the population of interest. We tend to think of the test data as the population of interest, but actually, what really matters is the data that will be seen at inference time. Evaluations assume that the test set is representative of what real users will use the system for, but that may or may not be the case. In practice, the test set is often very similar to the training set, probably more similar than either are to what real users are likely to expect from a real product. This is especially likely to be the case if test sets and training sets are collected before the product is shipped, and the world changes after the product ships and before users have a chance to use the product.

It is important to match test and training data as much as possible to application scenarios. Consider medical applications and BioBERT (Lee *et al.* 2020). BioBERT is more effective than BERT on a number of medical benchmarks, probably because BioBERT was trained on data<sup>17</sup> that are more appropriate for those use cases.

In short, it is often worth the effort to train on data that are as representative as possible to the population of interest, namely inputs that likely to be seen at inference time.

#### 1.4 Amortizing large upfront pre-training costs over many apps

Factoring the training task into pre-training and fine-tuning makes it possible to amortize large upfront investments in pre-training over many use cases. The business case for factoring is attractive because fine-tuning is relatively inexpensive compared to pre-training.

PTMs are somewhat similar to general purpose CPU chips. In the early days of Intel, they had a number of customers that wanted special purpose chips. Rather than designing a different custom chip for each customer, Intel invested in general purpose technology that could be customized as needed for many different use cases. In retrospect, the wisdom of general purpose solutions seems self-evident, though it was far from obvious at the time (Faggin 2009).

#### 1.5 Costs

This paper will focus on fine-tuning, which is more demanding than inference but less demanding than pre-training base models, in terms of both programming prerequisites and computational resources. As for computational resources, inference can be done without GPUs, fine-tuning requires at least one GPU, and pre-training is typically performed on a large cluster of GPUs. As for speed, inference takes seconds/minutes, fine-tuning takes hours, and pre-training takes days/weeks.

Costs tend to increase with both the size of the model and the size of the data. Current base models are large in both respects, as shown in Tables 1–2. Going forward, we can expect base models to become even larger in the future (Hestness *et al.* 2017; Kaplan *et al.* 2020).

<sup>16</sup><http://www.natcorp.ox.ac.uk/>.

<sup>17</sup><https://pubmed.ncbi.nlm.nih.gov/>.

Most base models in the literature were trained in well-funded industrial laboratories that have access to large GPU clusters, and large datasets. Training base models may be too expensive for universities and start-up companies as discussed in Section 1.3 of Bommasani *et al.* (2021):<sup>18</sup>

*[R]esearch on building foundation [base] models themselves has occurred almost exclusively in industry — big tech companies such as Google, Facebook, Microsoft, or Huawei*

The cost of training base models came up several times at a recent Stanford Workshop on Foundation Models;<sup>19,20</sup> academics may not have the means to train base models, especially if costs continue to escalate.

In addition to concerns in academia, there are also concerns with costs in industry. Larger models are expensive to run in the cloud, and infeasible at the edge (in a phone). These concerns have motivated work on compression methods (Hinton, Vinyals, and Dean 2015; Howard *et al.* 2017; Cheng *et al.* 2017; Gupta and Agrawal 2020; Su *et al.* 2021). These methods compress a large model down to a smaller one. In hubs, these smaller models are often labeled as tiny and/or mobile. It appears that the best way to train a tiny mobile model is to train a large model and then compress it. One might have thought that one could get to a better place directly, by training the tiny model in the obvious way, but experience suggests that indirect route via a larger model is more effective than the direct route.

Compression methods may also play an important role for energy policies and global warming. There has been some concern about energy consumption during training (Strubell, Ganesh, and McCallum 2019), but actually, energy consumption during training is relatively small compared to energy consumption at inference time. If a billion users use your speech recognition model every day for a few years, then the one-time upfront investments to train the model are a round-off error compared to the recurring costs of running the model in production. Assuming that energy costs (and all other costs) scale with the size of the model, then the size of the model in production is far more important than the size of the model during training. Thus, we should not feel too guilty about training a large model, as long as we compress the model before we ship it to a billion users.

### 1.6 Further reading

It is hard to keep up with all the recent progress. Performance on benchmarks keeps going up and up and up. Many of the papers are very recent. In addition to the papers cited in this paper, an excellent overview of recent progress can be found here.<sup>21</sup> There are also many excellent blogs, tutorials, and surveys on fine-tuning and pre-training<sup>22,23,24</sup> (Ruder *et al.* 2019; Han *et al.* 2021).

## 2. Examples of fine-tuning

This section will discuss three examples of fine-tuning. Code is based on PaddleHub and HuggingFaceHub.

### 2.1 Flowers: An example of fine-tuning

As mentioned in Section 1.2, fine-tuning is “unreasonably effective” in transferring knowledge from pre-trained base models (ResNet for ImageNet) to novel tasks (flower classification). This

<sup>18</sup><https://crfm.stanford.edu/workshop.html>.

<sup>19</sup><https://crfm.stanford.edu/workshop.html>.

<sup>20</sup><https://youtu.be/RLrjKGN89Fc?t=818>.

<sup>21</sup><https://github.com/thunlp/PLMpapers>.

<sup>22</sup><http://tiny.cc/NAACLTransfer>.

<sup>23</sup>[https://www.tensorflow.org/tutorials/images/transfer\\_learning](https://www.tensorflow.org/tutorials/images/transfer_learning).

<sup>24</sup>[https://keras.io/guides/transfer\\_learning/](https://keras.io/guides/transfer_learning/).

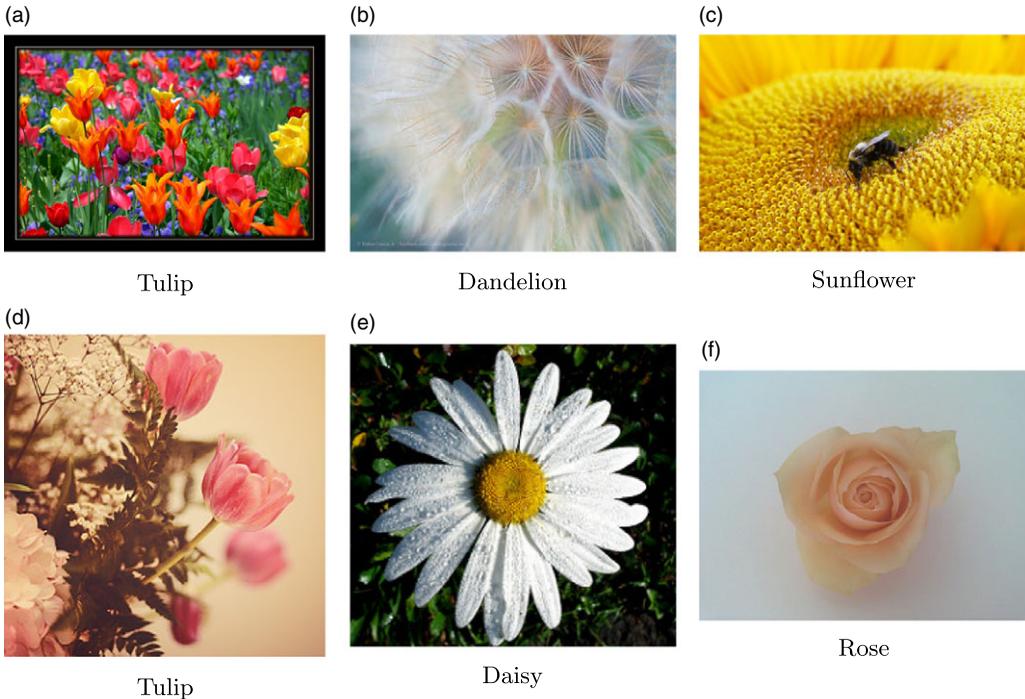


Figure 1. Some pictures of flowers with labels.

subsection will discuss a simplified version the 102 Category Flower Dataset<sup>25</sup> (Nilsback and Zisserman 2008). Performance on that task continues to improve.<sup>26</sup> Top performing systems (Kolesnikov *et al.* 2020) are using fine-tuning, as well as more data and more ideas. The simplified version uses 5 classes as opposed to 102.

There are a number of tutorials that explain how to fine-tune a base model to classify flowers.<sup>27</sup> We have posted some code for this example on GitHub.<sup>28</sup> Our code is based on an example from PaddleHub.<sup>29</sup>

These examples start with ResNet<sup>30</sup> (He *et al.* 2016) as a pre-trained base model. The base model was trained on 14M images and 1000 classes from ImageNet (Deng *et al.* 2009). The fine-tuning task is to modify the base model to recognize 5 types of flowers instead of the 1000 ImageNet classes. The 5 flower classes are: *rose*, *tulip*, *daisy*, *sunflower*, and *dandelion*. Six examples of flowers and class labels are shown in Figure 1.

For fine-tuning, we are given a training set and a validation set. Both sets consist of pictures of flowers,  $x$ , labeled with the 5 classes,  $y$ . There are 2915 flowers in the training set, and 383 flowers in the validation set. The validation set is used to measure error. That is, after fine-tuning, the model is given a picture from the validation set,  $x$ , and asked to predict a label,  $\hat{y}$ . These predictions,  $\hat{y}$ , are compared with gold labels,  $y$ , to produce a score.

At inference/evaluation time, we are given a novel picture,  $x$ , and a set of possible class labels such as the 5 classes of flowers. The model predicts a label,  $\hat{y}$ , one of the class labels.

<sup>25</sup><https://www.robots.ox.ac.uk/vgg/data/flowers/102/>.

<sup>26</sup><https://paperswithcode.com/sota/fine-grained-image-classification-on-oxford>.

<sup>27</sup>[https://www.tensorflow.org/hub/tutorials/image\\_feature\\_vector#the\\_flowers\\_dataset](https://www.tensorflow.org/hub/tutorials/image_feature_vector#the_flowers_dataset).

<sup>28</sup>[https://github.com/kwchurch/deepnet\\_examples/tree/main/fine-tuning/examples/PaddleHub/image\\_classification](https://github.com/kwchurch/deepnet_examples/tree/main/fine-tuning/examples/PaddleHub/image_classification).

<sup>29</sup>[https://github.com/PaddlePaddle/PaddleHub/tree/release/v2.1/demo/image\\_classification](https://github.com/PaddlePaddle/PaddleHub/tree/release/v2.1/demo/image_classification).

<sup>30</sup><https://towardsdatascience.com/understanding-and-coding-a-resnet-in-keras-446d7ff84d33>.

Before fine-tuning, the model is performing at chance since the ImageNet uses different classes. After fine-tuning, the model is considerably better than chance, though far from state of the art (SOTA).

Fine-tuning is just one of many tools in the toolbox. If one wants to top the leaderboard, one needs an “unfair advantage,” something better than what the competition is likely to do. Since fine-tuning is now well established within the literature, one should assume that the competition is likely to do that. One is unlikely to do much better than the competition (or much worse than the competition) if one uses obvious methods (such as fine-tuning) in obvious ways.

## 2.2 SQuAD: Another example of fine-tuning

SQuAD 1.1 (Stanford Question Answering Dataset) (Rajpurkar *et al.* 2016) and SQuAD 2.0 (Rajpurkar, Jia, and Liang 2018) are a popular benchmark for question answering (Q&A). Several solutions are posted on GitHub.<sup>31</sup> These are all very short (less than 40 lines) and easy to read. There are many blogs<sup>32</sup> and videos discussing inference with BERT-like models that have been fine-tuned for SQuAD. This video has 32k views on YouTube.<sup>33</sup> The large number of views makes it clear that many people want to know how to do this.

What is the SQuAD task? The inputs,  $x$ , are questions and short documents (containing no more than 512 subword tokens). Outputs,  $y$ , are answers. By construction, answers are spans, substrings of the input documents. For example:

- Input Document: *The American Football Conference (AFC) champion Denver Broncos defeated the National Football Conference (NFC) champion Carolina Panthers 24–10 to earn their third Super Bowl title.*
- Input Question: *What does AFC stand for?*
- Output Gold Answer: *American Football Conference*

The solutions above download a model that has already been fine-tuned for SQuAD. Suppose we want to fine-tune our own model. How can we do that? We have posted a short shell script on GitHub<sup>34</sup> based on an example from HuggingFaceHub.<sup>35</sup> HuggingFaceHub’s page reports F1 of 88.52 and exact\_match accuracy of 81.22. We ran our shell script five times and obtained slightly worse results: F1 of  $88.45 \pm 0.020$ , and exact\_match of  $81.2 \pm 0.055$ . Standard deviations are small but non-zero, because results vary slightly from one run to another. If you run these scripts, your results are likely to be similar, though not exactly the same as what is reported here.

Our script is borrowed from the first (and simplest) solution on the HuggingFaceHub page in footnote<sup>35</sup>. That page provides a number of variations of this solution. The last solution on that page is considerably better in terms of F1 (93.52). There are a number of factors that contribute to these improvements, including: (a) better pre-trained base models, (b) better hyperparameters, and (c) more GPUs.

Table 3 shows the numbers reported above, as well as one more row for the SOTA solution (Yamada *et al.* 2020) from two leaderboards: (a) SQuAD 1.1<sup>36</sup> and (b) papers with code.<sup>37</sup>

<sup>31</sup>[https://github.com/kwchurch/deepnet\\_examples/blob/main/pretrained/examples/PaddleNLP/inference/question\\_answering/](https://github.com/kwchurch/deepnet_examples/blob/main/pretrained/examples/PaddleNLP/inference/question_answering/).

<sup>32</sup><https://mccormickml.com/2020/03/10/question-answering-with-a-fine-tuned-BERT/>

<sup>33</sup><https://www.youtube.com/watch?v=l8ZYCvgGu0o>.

<sup>34</sup>[https://github.com/kwchurch/deepnet\\_examples/tree/main/fine-tuning/examples/HuggingFaceHub/question-answering](https://github.com/kwchurch/deepnet_examples/tree/main/fine-tuning/examples/HuggingFaceHub/question-answering).

<sup>35</sup><https://github.com/huggingface/transformers/tree/master/examples/pytorch/question-answering>.

<sup>36</sup><https://rajpurkar.github.io/SQuAD-explorer/>.

<sup>37</sup><https://paperswithcode.com/sota/question-answering-on-squad11>.

**Table 3.** Some SQuAD 1.1 results

Method	F1	Exact match
HuggingFaceHub first solution	88.52	81.22
Our replication	88.45 ± 0.020	81.2 ± 0.055
HuggingFaceHub last solution	93.52	
Human performance	91.221	82.304
SQuAD 1.1 leaderboard (Yamada <i>et al.</i> 2020)	<b>95.719</b>	<b>90.622</b>

Estimates for human performance from Rajpurkar *et al.* (2016) are posted on the SQuAD 1.1 leaderboard.

### 2.2.1 Alchemy, unrealistic expectations, and other concerns

The simple description of fine-tuning sounds simple, though there are many details that can make a big difference in practice for reasons that are not well understood. NeurIPS (formerly NIPS) used to be more rigorous and more theoretical, but these days, the practice is well ahead of the theory. Ali Rahimi and Ben Recht suggested that “machine learning has become alchemy” in a controversial test-of-time-award talk at NIPS-2017.<sup>38</sup>

There have been many booms and busts in AI over the decades (Church 2011). Expectations tend to increase during booms, inevitably leading to the next bust. There can be benefits to hyping our successes, especially in the short-term, but in the long-term, it is risky (and counter-productive) to set unrealistic expectations.

While fine-tuning performance is impressive on the SQuAD benchmark, and may exceed human performance in certain evaluations, we should not fool ourselves into believing that we have accomplished more than we have:<sup>39</sup>

*Microsoft researchers have created technology that uses artificial intelligence to read a document and answer questions about it about as well as a human. . . . Ming Zhou, assistant managing director of Microsoft Research Asia, said the SQuAD dataset results are an important milestone, but. . . people are still much better than machines at comprehending . . . language. . . This milestone is just a start.*<sup>40</sup>

Based on successes such as SQuAD, the community may believe that modern methods based on BERT will out-perform older rule-based systems, at least for SQuAD-like questions. However, for special cases of SQuAD questions such as acronyms: AFC (American Football Conference), we found that a rule-based solution, Ab3P (Sohn *et al.* 2008), is better than BERT (Liu *et al.* 2021). BERT-type models may not be the best way to capture constraints on spelling/sound such as acronyms, puns, alliteration, rhymes, and meter (e.g., iambic pentameter).

While fine-tuning works well on simple factoid questions in SQuAD, it is not clear what will be needed to deal with real questions in such search logs such as those in Table 4 from the DuReader benchmark<sup>41</sup> (He *et al.* 2018).

<sup>38</sup><http://www.argmin.net/2017/12/05/kitchen-sinks/>.

<sup>39</sup><https://www.azquotes.com/quote/95368?ref=fooling-yourself>.

<sup>40</sup><https://blogs.microsoft.com/ai/microsoft-creates-ai-can-read-document-answer-questions-well-person/>.

<sup>41</sup><https://github.com/baidu/DuReader>.

**Table 4.** English glosses of six types of questions from Chinese search logs

	Fact	Opinion
Entity	On which day will iPhone be released	Top 10 movies of 2017
Description	Why are firetrucks red	How good is Toyota Corolla
Yes-No	Is 39.5 degree a high fever	Does learning to play go improve intelligence

**Table 5.** GLUE Results for Human, HuggingFaceHub and Our replication

	CoLA	SST-2	MRPC	STS-B	QQP
Hum	66.4	97.8	80.8/86.3	92.7/92.6	80.4/59.5
Hug	56.53	92.32	88.85/84.07	88.64/88.48	90.71/87.49
Rep	55.79	91.86	89.76/85.29	89.26/88.93	90.80/87.58
	MNLI	QNLI	RTE	WNLI	
Hum	92.0/92.8	91.2	93.6	95.9	
Hug	83.91/84.10	90.66	65.70	56.34	
Rep	84.31	90.74	65.72	33.80	

### 2.3 GLUE: Yet another example of fine-tuning

Our final example of fine-tuning is the GLUE benchmark (Wang *et al.* 2018). The benchmark consists of 9 tasks, as shown in Table 5. There are 9 training sets and 9 test sets for each of the 9 tasks. Fine-tuning combines a base model such as BERT and a training set to produce a model. In this way, we construct 9 models and evaluate them on the 9 test sets. The 9 tasks in the GLUE benchmark use different metrics, as documented in the HuggingFace tutorial for fine-tuning for GLUE.<sup>42</sup>

As for evaluation, many of the tasks score by accuracy, though some score by F1 and correlation. In all cases, larger numbers are better. Performance is reported on the leaderboard.<sup>43</sup> SuperGLUE<sup>44</sup> (Wang *et al.* 2019) is an updated version of GLUE.<sup>45</sup>

Leaderboards rank systems by average performance. It may not be appropriate to average metrics of performance based on incompatible scales: accuracy, F1, and correlation.

At a recent ACL-2021 workshop on Benchmarking,<sup>46</sup> John Mashey suggested replacing arithmetic means with geometric means, based on his experience with the SPEC benchmark, an important benchmark for measuring CPU performance over the last few decades<sup>47</sup> (Mashey 2005).

Performance on GLUE leaderboards is currently close to human performance.<sup>48</sup> Estimates of human performance in Table 5 are based on Nangia and Bowman (2019).

<sup>42</sup><https://github.com/huggingface/transformers/tree/master/examples/pytorch/text-classification>.

<sup>43</sup><https://gluebenchmark.com/>.

<sup>44</sup><https://super.gluebenchmark.com/>.

<sup>45</sup><https://www.youtube.com/watch?v=1CoLguOswLo>.

<sup>46</sup>[https://github.com/kwchurch/Benchmarking\\_past\\_present\\_future/blob/master/README.md#Mashey](https://github.com/kwchurch/Benchmarking_past_present_future/blob/master/README.md#Mashey).

<sup>47</sup><https://www.spec.org/>.

<sup>48</sup><http://research.baidu.com/Blog/index-view?id=160>.

The Hug scores in Table 5 are from the URL in footnote 42. Our replication uses a short shell script<sup>49</sup> that also makes use of the URL in footnote 42. According to the HuggingFace documentation, all of these fine-tuning tasks can be completed in a few hours on a single GPU. Our replications were run on a single GPU, and none of the tasks took more than a few hours.

The names of the tasks are not that useful for understanding what these tasks are measuring. We find the instructions to the human annotators<sup>50</sup> to be helpful for this purpose. The GLUE tasks tend to reflect the particular interests of the researchers that created the benchmark, and may not be as representative of the larger field of natural language processing.

### 2.3.1 Winograd schema

Of the 9 tasks, WNLI (Winograd Schema) is the most challenging. The most frequent label baseline is about 65% accurate on this task. Note that both Hug and Rep in Table 5 are well below this baseline. Following Devlin *et al.* (2019), many authors exclude this task because this task is so hard, and also because this task may not be suitable for fine-tuning.

Winograd Schema was designed to be hard for machines but easy for people (Levesque, Davis, and Morgenstern 2012). There was a Winograd Schema Challenge at IJCAI-16, where no system did better than chance (Davis, Morgenstern, and Ortiz 2017). The fine-tuning recipe described in this paper is no better than most-frequent-class guessing (65%). Until recently, most systems were no better than guessing, but now, the systems at the top of the leaderboard achieve about 90% on WNLI, closer to human performance of 96% (Nangia and Bowman 2019). It is not clear how these systems reach that impressive level of performance, though there are a few recent papers that provide some clues (Liu *et al.* 2019; Sakaguchi *et al.* 2020).

## 3. GPT-3 teaser

Recently, there has been quite a bit of excitement surrounding GPT-3. GPT-3 is a large PTM, but unlike the other models in Table 1, GPT-3 is probably not suitable for fine-tuning. In addition, access to GPT-3 is limited to a relatively small set of people that have been granted permission to use the beta version of the API.

GPT-3 is amazingly fluent, but not credible enough for fact checking:

*To the extent that a use case places importance on the truth of the outputs provided, it is not a good fit for GPT-3.* (Dale 2021)

Following a suggestion by Percy Liang<sup>51</sup> at a recent workshop at Stanford,<sup>52</sup> we gave GPT-3 the prompt: “In 1885, Stanford University was. . .” GPT-3 produced the following:

- **In 1885, Stanford University was** founded by Leland and Jane Stanford in memory of their only child, recently deceased.
- In 1887, the first successful gasoline powered automobile in the United States was manufactured in Cleveland, Ohio by Charles and Frank Duryea.
- In 1887, the American Federation of Labor was founded.
- In 1887, the Salvation Army was formally incorporated in London, England.
- In 1890, the American Red Cross was established by Clara Barton.
- In 1891, the Salvation Army in the United States was founded.
- In 1892, the Boston Red Sox baseball team was established.

<sup>49</sup>[https://github.com/kwchurch/deepnet\\_examples/tree/main/fine-tuning/examples/HuggingFaceHub/question-answering](https://github.com/kwchurch/deepnet_examples/tree/main/fine-tuning/examples/HuggingFaceHub/question-answering).

<sup>50</sup><https://nyu-ml.github.io/GLUE-human-performance/>.

<sup>51</sup><https://youtu.be/dG628PEN1fY?t=365>.

<sup>52</sup><https://crfm.stanford.edu/workshop.html>.

- In 1893, the first Miss America Pageant was held in New York City.
- In 1895, the National Association for the Advancement of Colored People (NAACP) was founded.
- In 1895, the National Council of Women of the United States was founded.
- In 1897, the Boston Symphony Orchestra gave its first performance.
- In 1897, the first American open tennis tournament was held at the Newport Casino in Rhode Island.
- In 1897, the first practical typewriter was patented by Christopher Sholes.
- In 1897, the first winner of the Nobel Peace Prize was born.
- In 1899, the first automobile race was held at Narragansett Park in Cranston, Rhode Island.
- In 1900, the first automobile race was held at Narragansett Park in Cranston, Rhode Island.
- In 1900, the National Association for the Advancement of Colored People (NAACP) was founded.

The output feels like a farce. The first few statements are reasonably plausible, especially when the prompt matches a Wikipedia page. But the argument quickly degenerates into contradictions, tautologies, repetitions, etc. Many of the dates are wrong. A few are off by decades.

There are concerns about bias (Abid, Farooqi, and Zou 2021). While there are plenty of awful pages on the web, we are even more concerned about GPT-3's tendency to (mis)-quote authors out-of-context without attribution. Misquoting seems like a more dangerous violation of ethics than repeating what someone actually said. GPT-3 has a tendency to make up "facts," and say them in a way that people might actually believe what it says.

There are quite a few blogs<sup>53</sup> that attempt to characterize what GPT-3 "understands" and what it does not "understand." A number of evaluations are currently under review. One can probably show that GPT-3 can add two small numbers,<sup>54</sup> but when asked to add two big numbers, or to compute more complicated expressions, it just makes up answers.

Timelines such as the one above should be in chronological order, but GPT-3 does not appear to understand time, space, negation, etc. Given observations like these, it has been suggested that GPT-3 might be more appropriate for generating fiction (and poetry) than nonfiction. But GPT-3 does not appear to master rhyming or meter.

There was another case where we kept asking GPT-3 to generate more and more output because the argument always seemed to be just about to get to the point. But then we realized that we were reading into the argument more than there was. The argument was never going to get to the point, because there was no point (or even a point of view). The argument was not going anywhere. There was no structure to the argument. In fact, it is probably a mistake to refer to output from GPT-3 as an "argument."

As suggested in Dale (2021), GPT-3 must be good for something. It is so amazingly fluent. One possible use case is to help language learners (and others) to write more fluent prose. With an appropriate user interface, it should be possible to provide authors with the opportunity to improve fluency without changing the content too much. In this way, it should be possible to address Pascale Fung's concerns about "safety" in her comments at the workshop mentioned in footnote 52.

#### 4. Conclusions

Performance on a number of leaderboards has improved dramatically because of advances in fine-tuning. This paper described three examples of fine-tuning: (a) flowers (Section 2.1), (b) SQuAD (Section 2.2), and (c) GLUE (Section 2.3). In all three cases, we provided a short shell script on

<sup>53</sup><https://www.gwern.net/GPT-3>.

<sup>54</sup>See Figure 3.10 of Brown *et al.* (2020).

GitHub.<sup>55</sup> These scripts point to tutorials on PaddleHub and HuggingFaceHub (see footnotes 2 and 3). Our replications have performance that is close to the performance reported in the tutorials.

There are many more use cases for fine-tuning. We have used fine-tuning in speech for classifying emotions and Alzheimer's disease (Yuan *et al.* 2021b; Yuan *et al.* 2021a). Fine-tuning is likely to become one of the more popular methods for an extremely wide range of use cases in many fields including computational linguistics, speech, and vision.

That said, while the performance of fine-tuning is impressive on many benchmarks and may exceed human performance in certain evaluations, we should not fool ourselves into believing that we have accomplished more than we have. There are good reasons for caution, though maybe not for the reasons suggested in Bender *et al.* (2021). The problem is not so much that the models are too big, or that they are just memorizing the training data, but we are concerned about “sucking the oxygen out of the room.” That is, while fine-tuning is addressing many important and practical problems that matter on many benchmarks, there are many other issues that are also important and need to be addressed such as time, space, negation, order, and structure. Fine-tuning is a very effective hammer because many tasks look like nails, but it is unlikely that fine-tuning is all we need, because not all tasks look like nails.

## References

- Abid A., Farooqi M. and Zou J. (2021). Persistent anti-muslim bias in large language models. arXiv preprint arXiv:2101.05783.
- Aston G. and Burnard L. (1998). *The BNC Handbook*. Edinburgh University Press, Edinburgh. See also the BNC homepage at <https://protect-eu.mimecast.com/s/TE-UCxvqqTZpyjIvWKBL?domain=natcorp.ox.ac.uk/>.
- Baevski A., Zhou H., Mohamed A. and Auli M. (2020). wav2vec 2.0: a framework for self-supervised learning of speech representations. arXiv preprint arXiv:2006.11477.
- Bender E.M., Gebru T., McMillan-Major A. and Shmitchell S. (2021). On the dangers of stochastic parrots: can language models be too big? In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, pp. 610–623.
- Blum A. and Mitchell T. (1998). Combining labeled and unlabeled data with co-training. In *Proceedings of the Eleventh Annual Conference on Computational Learning Theory*, pp. 92–100.
- Bommasani R., Hudson D.A., Adeli E., Altman R., Arora S., von Arx S., Bernstein M.S., Bohg J., Bosselut A., Brunskill E., Brynjolfsson E., Buch S., Card D., Castellon R., Chatterji N., Chen A., Creel K., Davis J.Q., Demszky D., Donahue C., Doumbouya M., Durmus E., Ermon S., Etchemendy J., Ethayarajh K., Fei-Fei L., Finn C., Gale T., Gillespie L., Goel K., Goodman N., Grossman S., Guha N., Hashimoto T., Henderson P., Hewitt J., Ho D.E., Hong J., Hsu K., Huang J., Icard T., Jain S., Jurafsky D., Kalluri P., Karamcheti S., Keeling G., Khani F., Khattab O., Kohd P. W., Krass M., Krishna R., Kuditipudi R., Kumar A., Ladhak F., Lee M., Lee T., Leskovec J., Levent I., Li X.L., Li X., Ma T., Malik A., Manning C.D., Mirchandani S., Mitchell E., Munyikwa Z., Nair S., Narayan A., Narayanan D., Newman B., Nie A., Nibbles J.C., Nilforoshan H., Nyarko J., Ogut G., Orr L., Papadimitriou I., Park J. S., Piech C., Portelance E., Potts C., Raghunathan A., Reich R., Ren H., Rong F., Roohani Y., Ruiz C., Ryan J., Ré C., Sadigh D., Sagawa S., Santhanam K., Shih A., Srinivasan K., Tamkin A., Taori R., Thomas A.W., TramÈr F., Wang R.E., Wang W., Wu B., Wu J., Wu Y., Xie S.M., Yasunaga M., You J., Zaharia M., Zhang M., Zhang T., Zhang X., Zhang Y., Zheng L., Zhou K. and Liang, P. (2021). On the opportunities and risks of foundation models. arXiv preprint arXiv:2108.07258.
- Brown T.B., Mann B., Ryder N., Subbiah M., Kaplan J., Dhariwal P., Neelakantan A., Shyam P., Sastry G., Askell A., Agarwal S., Herbert-Voss A., Krueger G., Henighan T., Child, R., Ramesh A., Ziegler D.M., Wu J., Winter C., Hesse C., Chen, M., Sigler E., Litwin M., Gray S., Chess B., Clark J., Berner C., McCandlish S., Radford A., Sutskever I. and Amodei D. (2020). Language models are few-shot learners. In *NeurIPS*.
- Buck C., Heafield K. and van Ooyen B. (2014). N-gram counts and language models from the common crawl. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC-2014)*, Reykjavik, Iceland. European Languages Resources Association (ELRA), pp. 3579–3584.
- Burnard L. (2002). *Where did we go Wrong? A Retrospective Look at the British National Corpus*. In Kettemann, B. and Markus, G. (eds), *Teaching and Learning by Doing Corpus Analysis*. Amsterdam: Rodopi, pp. 51–71.
- Callison-Burch C. (2009). Fast, cheap, and creative: evaluating translation quality using Amazon's Mechanical Turk. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, Singapore. Association for Computational Linguistics, pp. 286–295.

<sup>55</sup>[https://github.com/kwchurch/deepnet\\_examples/tree/main/fine-tuning](https://github.com/kwchurch/deepnet_examples/tree/main/fine-tuning).

- Chelba C., Mikolov T., Schuster M., Ge Q., Brants T., Koehn P. and Robinson T. (2013). One billion word benchmark for measuring progress in statistical language modeling. arXiv preprint arXiv:1312.3005.
- Chen X., Hsieh C.-J. and Gong B. (2021). When vision transformers outperform resnets without pretraining or strong data augmentations. arXiv preprint arXiv:2106.01548.
- Cheng Y., Wang D., Zhou P. and Zhang T. (2017). A survey of model compression and acceleration for deep neural networks. arXiv preprint arXiv:1710.09282.
- Church K. (2011). A pendulum swung too far. *Linguistic Issues in Language Technology* 6(5), 1–27.
- Church K. and Bian Y. (2021). Data collection vs. knowledge graph completion: what is needed to improve coverage? In *EMNLP*.
- Church K.W., Yuan X., Guo S., Wu Z., Yang Y. and Chen Z. (2021). Emerging trends: deep nets for poets. *Natural Language Engineering* 27(5), 631–645.
- Conneau A., Khandelwal K., Goyal N., Chaudhary V., Wenzek G., Guzmán F., Grave E., Ott M., Zettlemoyer L. and Stoyanov V. (2020). Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Online. Association for Computational Linguistics, pp. 8440–8451.
- Dagan I., Glickman O. and Magnini B. (2005). The pascal recognising textual entailment challenge. In *Machine Learning Challenges Workshop*. Springer, pp. 177–190.
- Dale R. (2021). Gpt-3: What's it good for? *Natural Language Engineering* 27(1), 113–118.
- Davis E., Morgenstern L. and Ortiz C.L. (2017). The first Winograd schema challenge at ijcai-16. *AI Magazine* 38(3), 97–98.
- Deng J., Dong W., Socher R., Li L.-J., Li K. and Fei-Fei L. (2009). Imagenet: a large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, pp. 248–255.
- Devlin J., Chang M.-W., Lee K. and Toutanova K. (2019). BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, Minneapolis, Minnesota. Association for Computational Linguistics, pp. 4171–4186.
- Dosovitskiy A., Beyer L., Kolesnikov A., Weissenborn D., Zhai X., Unterthiner T., Dehghani M., Minderer M., Heigold G., Gelly S., Uszkoreit J. and Houshy N. (2021). An image is worth 16x16 words: transformers for image recognition at scale. In *ICLR*.
- Du J., Na X., Liu X. and Bu H. (2018). Aishell-2: transforming mandarin ASR research into industrial scale. arXiv preprint arXiv:1808.10583.
- Faggin F. (2009). The making of the first microprocessor. *IEEE Solid-State Circuits Magazine* 1(1), 8–21.
- Finin T., Murnane W., Karandikar A., Keller N., Martineau J. and Dredze M. (2010). Annotating named entities in twitter data with crowdsourcing. In *Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*, Los Angeles. Association for Computational Linguistics, pp. 80–88.
- Francis W.N. and Kučera H. (1979). Brown corpus manual. *Letters to the Editor* 5(2), 7.
- Francis W.N. and Kučera H. (1982). *Frequency Analysis of English Usage: Lexicon and Grammar*, Boston, MA, USA. Houghton Mifflin.
- Gupta M. and Agrawal P. (2020). Compression of deep learning models for text: a survey. arXiv preprint arXiv:2008.05221.
- Han X., Zhang Z., Ding N., Gu Y., Liu X., Huo Y., Qiu J., Zhang L., Han W., Huang M., Jin Q., Lan Y., Liu Y., Liu Z., Lu Z., Qiu X., Song R., Tang J., Wen J., Yuan J., Zhao W.X. and Zhu J. (2021). Pre-trained models: past, present and future. CoRR, abs/2106.07139.
- Hara K., Adams A., Milland K., Savage S., Callison-Burch C. and Bigham J.P. (2018). A data-driven analysis of workers' earnings on amazon mechanical turk. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, pp. 1–14.
- Hayashi T., Yamamoto R., Inoue K., Yoshimura T., Watanabe S., Toda T., Takeda K., Zhang Y. and Tan X. (2020). Espnet-TTS: unified, reproducible, and integratable open source end-to-end text-to-speech toolkit. In *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, pp. 7654–7658.
- He K., Zhang X., Ren S. and Sun J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778.
- He W., Liu K., Liu J., Lyu Y., Zhao S., Xiao X., Liu Y., Wang Y., Wu H., She Q., Liu X., Wu T. and Wang H. (2018). DuReader: a Chinese machine reading comprehension dataset from real-world applications. In *Proceedings of the Workshop on Machine Reading for Question Answering*, Melbourne, Australia. Association for Computational Linguistics, pp. 37–46.
- Hestness J., Narang S., Ardalani N., Diamos G., Jun H., Kianinejad H., Patwary M., Ali M., Yang Y. and Zhou Y. 2017. Deep learning scaling is predictable, empirically. arXiv preprint arXiv:1712.00409.
- Hinton G., Vinyals O. and Dean J. (2015). Distilling the knowledge in a neural network. arXiv preprint arXiv:1503.02531.
- Howard A.G., Zhu M., Chen B., Kalenichenko D., Wang W., Weyand T., Andreetto M. and Adam H. (2017). Mobilenets: efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861.

- Howard J. and Ruder S.** (2018). Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Melbourne, Australia. Association for Computational Linguistics, pp. 328–339.
- Inaguma H., Kiyono S., Duh K., Karita S., Yalta N., Hayashi T. and Watanabe S.** (2020). ESPnet-ST: all-in-one speech translation toolkit. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, Online. Association for Computational Linguistics, pp. 302–311.
- Kaplan J., McCandlish S., Henighan T., Brown T.B., Chess B., Child R., Gray S., Radford A., Wu J. and Amodei D.** (2020). Scaling laws for neural language models. arXiv preprint arXiv:2001.08361.
- Kolesnikov A., Beyer L., Zhai X., Puigcerver J., Yung J., Gelly S. and Houlsby N.** (2020). Big transfer (bit): general visual representation learning. In *Computer Vision—ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part V 16*. Springer, pp. 491–507.
- Kučera H. and Francis W.N.** (1967). *Computational Analysis of Present-Day American English*. Providence, RI, USA: Brown University Press.
- Lee J., Yoon W., Kim S., Kim D., Kim S., So C.H. and Kang J.** (2020). Biobert: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics* **36**(4), 1234–1240.
- Levesque H., Davis E. and Morgenstern L.** (2012). The Winograd schema challenge. In *Thirteenth International Conference on the Principles of Knowledge Representation and Reasoning*.
- Li C., Shi J., Zhang W., Subramanian A.S., Chang X., Kamo N., Hira M., Hayashi T., Boeddeker C., Chen Z. and Watanabe S.** (2021). ESPnet-SE: end-to-end speech enhancement and separation toolkit designed for ASR integration. In *Proceedings of IEEE Spoken Language Technology Workshop (SLT)*. IEEE, pp. 785–792.
- Liu B., Huang J., Cai X. and Church K.** (2021). Better than BERT but worse than baseline. arXiv preprint arXiv:2105.05915.
- Liu Y., Ott M., Goyal N., Du J., Joshi M., Chen D., Levy O., Lewis M., Zettlemoyer L., and Stoyanov V.** (2019). Roberta: a robustly optimized BERT pretraining approach.
- Mashey J.** (2005). Summarizing performance is no mean feat [computer performance analysis]. In *IEEE International. 2005 Proceedings of the IEEE Workload Characterization Symposium, 2005*, p. 1.
- Nangia N. and Bowman S.R.** (2019). Human vs. muppet: a conservative estimate of human performance on the GLUE benchmark. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, Florence, Italy. Association for Computational Linguistics, pp. 4566–4575.
- Nilsback M.-E. and Zisserman A.** (2008). Automated flower classification over a large number of classes. In *Indian Conference on Computer Vision, Graphics and Image Processing*.
- Ott M., Edunov S., Baevski A., Fan A., Gross S., Ng N., Grangier D. and Auli M.** (2019). fairseq: a fast, extensible toolkit for sequence modeling. In *Proceedings of NAACL-HLT 2019: Demonstrations*.
- Pan S.J. and Yang Q.** (2009). A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering* **22**(10), 1345–1359.
- Panayotov V., Chen G., Povey D. and Khudanpur S.** (2015). Librispeech: an ASR corpus based on public domain audio books. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, pp. 5206–5210.
- Pavlick E., Post M., Irvine A., Kachaev D., and Callison-Burch C.** (2014). The language demographics of Amazon mechanical turk. *Transactions of the Association for Computational Linguistics* **2**, 79–92.
- Radford A., Wu J., Child R., Luan D., Amodei D. and Sutskever I.** (2019). Language models are unsupervised multitask learners. OpenAI Blog.
- Rajpurkar P., Jia R. and Liang P.** (2018). Know what you don't know: unanswerable questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, Melbourne, Australia. Association for Computational Linguistics, pp. 784–789.
- Rajpurkar P., Zhang J., Lopyrev K. and Liang P.** (2016). SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, Austin, Texas. Association for Computational Linguistics, pp. 2383–2392.
- Rapp R.** (2014a). Using collections of human language intuitions to measure corpus representativeness. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, Dublin, Ireland. Dublin City University and Association for Computational Linguistics, pp. 2117–2128.
- Rapp R.** (2014b). Using word familiarities and word associations to measure corpus representativeness. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, Reykjavik, Iceland. European Language Resources Association (ELRA), pp. 2029–2036.
- Ravanelli M., Parcollet T., Plantinga P., Rouhe A., Cornell S., Lugosch L., Subakan C., Dawalatabad N., Heba A., Zhong J. et al.** (2021). Speechbrain: a general-purpose speech toolkit. arXiv preprint arXiv:2106.04624.
- Ruder S., Peters M.E., Swayamdipta S. and Wolf T.** (2019). Transfer learning in natural language processing. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Tutorials*, pp. 15–18.
- Sakaguchi K., Le Bras R., Bhagavatula C. and Choi Y.** (2020). Winogrande: an adversarial Winograd schema challenge at scale. In *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34(05), pp. 8732–8740.

- Schick T. and Schütze H. (2021). Exploiting cloze-questions for few-shot text classification and natural language inference. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, Online. Association for Computational Linguistics, pp. 255–269.
- Sohn S., Comeau D.C., Kim W. and Wilbur W.J. (2008). Abbreviation definition identification based on automatic precision estimates. *BMC Bioinformatics* 9(1), 402.
- Steiner A., Kolesnikov A., Zhai X., Wightman R., Uszkoreit J. and Beyer L. (2021). How to train your ViT? data, augmentation, and regularization in vision transformers. arXiv preprint arXiv:2106.10270.
- Strubell E., Ganesh A. and McCallum A. (2019). Energy and policy considerations for deep learning in NLP. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, Florence, Italy. Association for Computational Linguistics, pp. 3645–3650.
- Su W., Chen X., Feng S., Liu J., Liu W., Sun Y., Tian H., Wu H. and Wang H. (2021). Ernie-tiny: a progressive distillation framework for pretrained transformer compression. arXiv preprint arXiv:2106.02241.
- Sun Y., Wang S., Feng S., Ding S., Pang C., Shang J., Liu J., Chen X., Zhao Y., Lu Y., et al. (2021). Ernie 3.0: large-scale knowledge enhanced pre-training for language understanding and generation. arXiv preprint arXiv:2107.02137.
- Sun Y., Wang S., Li Y., Feng S., Tian H., Wu H. and Wang H. (2020). Ernie 2.0: a continual pre-training framework for language understanding. In *AAAI*.
- Tolstikhin I., Houlsby N., Kolesnikov A., Beyer L., Zhai X., Unterthiner T., Yung J., Steiner A., Keysers D., Uszkoreit J., Lucic M. and Dosovitskiy A. (2021). Mlp-mixer: an all-mlp architecture for vision. arXiv preprint arXiv:2105.01601.
- Wang A., Pruksachatkun Y., Nangia N., Singh A., Michael J., Hill F., Levy O. and Bowman S. (2019). Superglue: a stickier benchmark for general-purpose language understanding systems. In Wallach H., Laroche H., Beygelzimer A., d'Alché-Buc F., Fox E. and Garnett R. (eds), *Advances in Neural Information Processing Systems*, vol. 32. Curran Associates, Inc.
- Wang A., Singh A., Michael J., Hill F., Levy O. and Bowman S. (2018). GLUE: a multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, Brussels, Belgium. Association for Computational Linguistics, pp. 353–355.
- Watanabe S., Hori T., Karita S., Hayashi T., Nishitoba J., Unno, Y., Enrique Yalta Soplin N., Heymann J., Wiesner, M., Chen N., Renduchintala A. and Ochiai T. (2018). ESPnet: end-to-end speech processing toolkit. In *Proceedings of Interspeech*, pp. 2207–2211.
- Wu B., Xu C., Dai X., Wan A., Zhang P., Yan Z., Tomizuka M., Gonzalez J., Keutzer K. and Vajda P. (2020). Visual transformers: token-based image representation and processing for computer vision. arXiv preprint arXiv:2006.03677.
- Yamada I., Asai A., Shindo H., Takeda H. and Matsumoto Y. (2020). LUKE: deep contextualized entity representations with entity-aware self-attention. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Online. Association for Computational Linguistics, pp. 6442–6454.
- Yuan J., Cai X., Bian Y., Ye Z. and Church K. (2021a). Pauses for detection of Alzheimer's disease. *Frontiers in Computer Science* 2, 57.
- Yuan J., Cai X., Zheng R., Huang L. and Church K. (2021b). The role of phonetic units in speech emotion recognition. arXiv preprint arXiv:2108.01132.
- Zhu X.J. (2005). Semi-supervised learning literature survey. University of Wisconsin-Madison Department of Computer Sciences.
- Zhu Y., Kiros R., Zemel R., Salakhutdinov R., Urtasun R., Torralba A. and Fidler S. (2015). Aligning books and movies: towards story-like visual explanations by watching movies and reading books. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 19–27.