HellaSwag: Can a Machine *Really* Finish Your Sentence?

Rowan Zellers[♠] **Ari Holtzman**[♠] **Yonatan Bisk**[♠] **Ali Farhadi**[♠♡] **Yejin Choi**[♠♡]
[♠]Paul G. Allen School of Computer Science & Engineering, University of Washington
[♡]Allen Institute for Artificial Intelligence
https://rowanzellers.com/hellaswag

Abstract

문장이 주어지면
뒷부분을 추론하여 선택하는 형태 SWAG

Recent work by Zellers et al. (2018) introduced a new task of *commonsense natural language inference*: given an event description such as "A woman sits at a piano," a machine must select the most likely followup: "She sets her fingers on the keys." With the introduction of BERT (Devlin et al., 2018), near human-level performance was reached. Does this mean that machines can perform human level commonsense inference?

In this paper, we show that commonsense inference still proves difficult for even state-of-the-art models, by presenting *HellaSwag*, a new challenge dataset. Though its questions are trivial for humans (>95% accuracy), state-of-the-art models struggle (<48%). We achieve this via Adversarial Filtering (AF), a data collection paradigm wherein a series of discriminators iteratively select an adversarial set of machine-generated wrong answers. AF proves to be surprisingly robust. The key insight is to scale up the length and complexity of the dataset examples towards a critical 'Goldilocks' zone wherein generated text is ridiculous to humans, yet often misclassified by state-of-the-art models.

Our construction of *HellaSwag*, and its resulting difficulty, sheds light on the inner workings of deep pretrained models. More broadly, it suggests a new path forward for NLP research, in which benchmarks co-evolve with the evolving state-of-the-art in an adversarial way, so as to present ever-harder challenges.

1 Introduction

Imagine a woman chasing a dog around outside, trying to give it a bath. What might happen next? Humans can read a narrative like this, shown in Figure 1, and connect it to a rich model of the world: the dog is currently dry and not soapy, and it actively doesn't want to be bathed. Thus, one

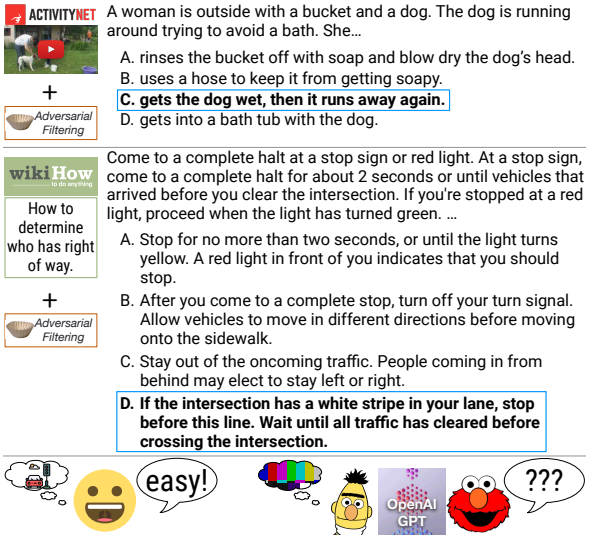

Figure 1: Models like BERT struggle to finish the sentences in *HellaSwag*, even when they come from the same distribution as the training set. While the wrong endings are on-topic, with words that relate to the context, humans consistently judge their meanings to be either incorrect or implausible. For example, option **A** of the WikiHow passage suggests that a driver should stop at a red light for **no more than two seconds**.

plausible next event is option **C**—that she'll get the dog wet and it will run away again.

When the SWAG dataset was first announced (Zellers et al., 2018), this new task of *commonsense natural language inference* seemed trivial for humans (88%) and yet challenging for then-state-of-the-art models (<60%), including ELMo (Peters et al., 2018). However, BERT (Devlin et al., 2018) soon reached over 86%, almost human-level performance. One news article on this development was headlined "*finally, a machine that can finish your sentence.*"[1]

In this paper, we investigate the following question: How well do deep pretrained models, like

[1]A New York Times article at https://nyti.ms/2DycutY.

BERT의 경우로 볼 때, Common Sense Reasoning이 잘되는 것 같지 않음
SWAG를 잘 푸는 이유는, 특정 데이터셋의 분포적인 편향성을 배우고 사용하기 때문임
이는 파인튜닝 단계에 의존적이며, 언어의 분포가 조금만 바뀌더라도 퍼포먼스가 급락하게 됨

BERT, perform at commonsense natural language inference (NLI)? Our surprising conclusion is that the underlying *task* remains unsolved. Indeed, we find that deep models such as BERT do not demonstrate robust commonsense reasonining ability by themselves. Instead, they operate more like *rapid surface learners* for a particular dataset. Their strong performance on SWAG is dependent on the finetuning process, wherein they largely learn to pick up on dataset-specific distributional biases. When the distribution of language shifts slightly, performance drops drastically – even if the domain remains identical.

We study this question by introducing *HellaSwag*,[2] a new benchmark for commonsense NLI. We use Adversarial Filtering (AF), a data-collection paradigm in which a series of discriminators is used to select a challenging set of generated wrong answers. AF is surprisingly effective towards this goal: the resulting dataset of 70k problems is easy for humans (95.6% accuracy), yet challenging for machines (<50%). This result holds even when models are given a significant number of training examples, and even when the test data comes from the exact same distribution as the training data. Machine performance slips an additional 5% when evaluated on examples that cover novel concepts from the same domain.

HellaSWAG는 AF를 사용하여 좀더 그럴듯한 오답지를 생성했음

To make this dataset robust to deep pretrained models, we use a trifecta of state-of-the-art generators (Radford et al., 2018), state-of-the-art discriminators (BERT), and high quality source text. We expand on the SWAG's original video-captioning domain by using WikiHow articles, greatly increasing the context diversity and generation length. Our investigation reveals a Goldilocks zone – roughly three sentences of context, and two generated sentences – wherein generations are largely nonsensical, even though state-of-the-art discriminators cannot reliably tell the difference between these generations and the ground truth.

More broadly, our paper presents a case-study towards a future of verified progress in NLP, via iterative rounds of building and breaking datasets. If our ultimate goal is to provide reliable benchmarks for challenging tasks, such as commonsense NLI, these benchmarks cannot be static. Instead, they must evolve together with the evolving state-of-the-art. Continued evolution in turn requires principled dataset creation algorithms. Whenever a new iteration of a dataset is created, these algorithms must leverage existing modeling advancements to filter out spurious biases. Only once this cycle becomes impossible can we say that the underlying *task* – as opposed an individual dataset – is solved.

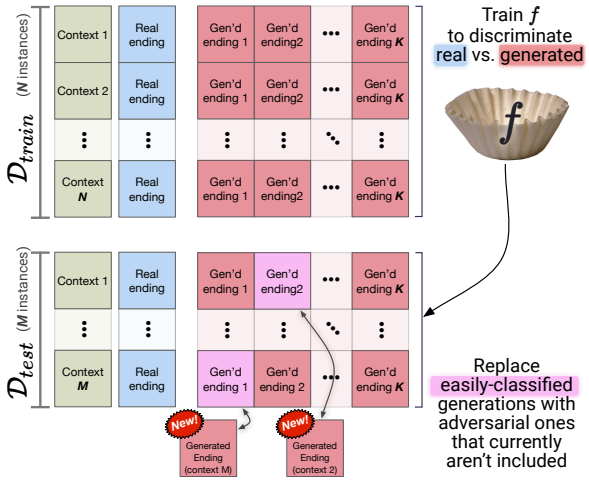

Figure 2: An overview of Adversarial Filtering. On each iteration, a new classifier is trained on a dummy training set $\mathcal{D}_{train}$ to replace easily-classified negative endings on the dummy test set $\mathcal{D}_{test}$ with adversarial endings. This process is repeated iteratively, to obtain a challenging dataset regardless of the final split.

2 Background

SWAG
비디오 지막을 Context로 받으면
다음에 이어질 4가지 상황중 하나를 선택

SWAG is a dataset for commonsense NLI. For each question, a model is given a **context** from a video caption and four **ending choices** for what might happen next. Only one choice is right – the actual next caption of the video.

Obtaining interesting negatives is challenging. Prior work (e.g. Gururangan et al., 2018; Poliak et al., 2018) has found that when humans write the endings to NLI questions, they introduce subtle yet strong class-conditional biases known as *annotation artifacts*.[3]

To address this, Zellers et al. (2018) introduced **Adversarial Filtering** (AF). An overview is shown in Figure 2. The key idea is to produce a dataset $\mathcal{D}$ which is adversarial for *any* arbitrary split of $(\mathcal{D}_{train}, \mathcal{D}_{test})$. This requires a *generator* of negative candidates (i.e., wrong endings that vi-

[2]Short for *Harder Endings, Longer contexts, and Low-shot Activities for Situations With Adversarial Generations*. Dataset and code at https://rowanzellers.com/hellaswag.

[3]These biases simply inflate model performance, but past work has also shown that are unwanted social biases induced when humans write the endings, in terms of gender and race (Rudinger et al., 2015).

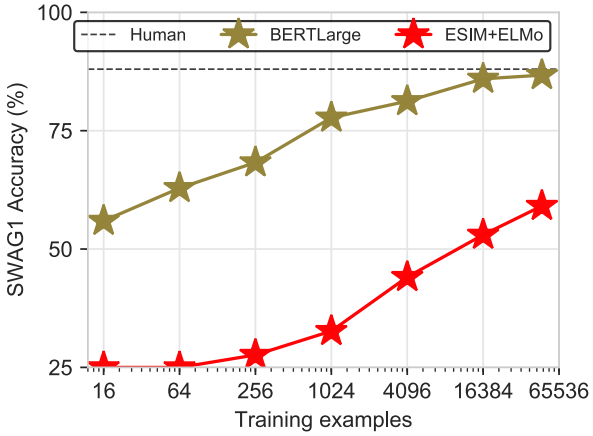

Figure 3: Validation accuracy on SWAG for BERT-Large versus training set size. The baseline (25% accuracy) is random chance. BERT does well given as few as 16 training examples, but requires tens of thousands of examples to approach human performance.

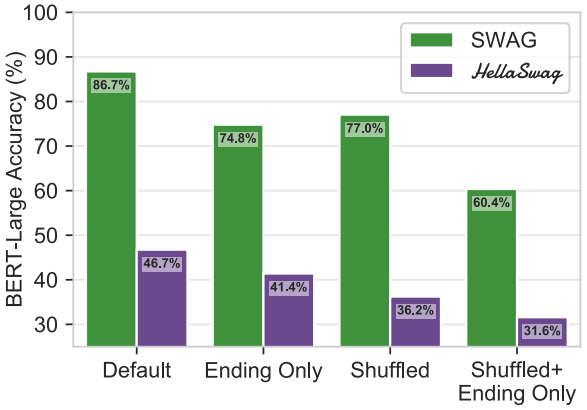

Figure 4: BERT validation accuracy when trained and evaluated under several versions of SWAG, with the new dataset *HellaSwag* as comparison. We compare:

`Ending Only`	No context is provided; just the endings.
`Shuffled`	Endings that are indidivually tokenized, shuffled, and then detokenized.
`Shuffled+ Ending Only`	No context is provided *and* each ending is shuffled.

olate human notions about how the world works), which we achieve by using a language model. Potential candidates of incorrect answers were massively oversampled from a language model trained on in-domain data, and then selected using an ensemble of adversaries. The selection process happens iteratively: on each iteration, the dataset is randomly partitioned into $\mathcal{D}_{train}$ and $\mathcal{D}_{test}$. The ensemble is trained to classify endings as *real* or *generated* on $\mathcal{D}_{train}$, then, AF replaces easy-to-classify generations in $\mathcal{D}_{test}$. This process continues until the accuracy of these adversaries converges. Last, humans validate the data to remove adversarial endings that seem realistic.

Importantly, AF creates a final dataset that is challenging to models regardless of the final dataset split. In Section 4, we will use AF as the underlying workhorse to construct an NLI dataset that is easy for humans, yet challenging for machines. This difficulty persists even when models are provided significant training data, and even when this data comes from the same distribution as the test set. This contrasts with past work on adversarial examples (e.g. Jia and Liang, 2017; Glockner et al., 2018; Belinkov and Bisk, 2018) which consider cases where an out-of-distribution test set is constructed to be adversarial.

3 Investigating SWAG

In this section, we investigate why SWAG was solved. We focus on BERT, since it is the best known approach at the time of writing.[4] Core to our analysis is investigating how a model trained on Wikipedia and books can be so effectively fine-tuned for SWAG, a dataset from video captions.

3.1 How much innate knowledge does BERT have about SWAG?

We investigate this question by measuring BERT's performance on SWAG while varying the size of the training dataset; results are shown in Figure 3. While the best known ELMo NLI model (ESIM+ELMo; Chen et al., 2017) requires the entire training set to reach 59%, BERT outperforms this given only 64 examples. However, BERT still needs upwards of 16k examples to approach human performance, around which it plateaus.

3.2 What is learned during finetuning?

Figure 4 compares BERT's performance when trained and evaluated on variants of SWAG.

Context: BERT's performance only slips 11.9 points (86.7%→74.8%) when context is omitted (`Ending Only`), suggesting a bias exists in the endings themselves.[5] If a followup event seems unreasonable *absent of context*, then there must be something markedly different between the space of human-written and machine-generated endings.

Structure: To distinguish word usage from

[4] See the appendix for a discussion of the BERT architecture and hyperparameter settings we used in our experiments.

[5] These biases are similar to those in NLI datasets, as found by Gururangan et al. (2018); Poliak et al. (2018).

BERT의 경우, Context를 생략하거나, Strcucture를 Shuffling하면 퍼포먼스가 많이 떨어짐
이는 모델이 어휘들의 배치에 대한 분포적 특성을 배우는 것으로 보임

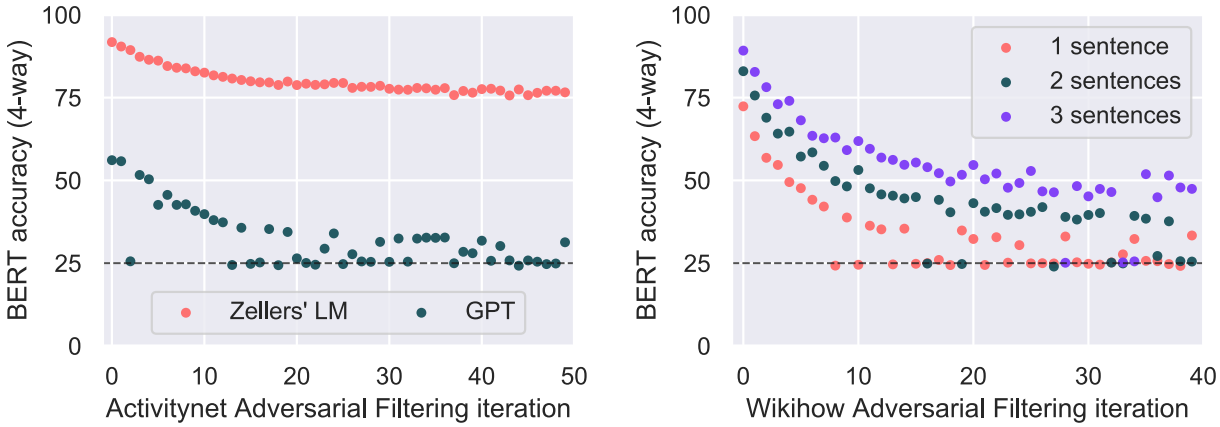

Figure 5: Adversarial Filtering (AF) results with BERT-Large as the discriminator. **Left**: AF applied to ActivityNet generations produced by Zellers et al. (2018)'s language model versus OpenAI GPT. While GPT converges at random, the LM used for SWAG converges at 75%. **Right**: AF applied to WikiHow generations from GPT, while varying the ending length from one to three sentences. They converge to random, ∼40%, and ∼50%, respectively.

structural patterns, we consider a new scenario, `Shuffled`. Here the shared context is provided, but the words in each ending choice are randomly permuted. Surprisingly, this reduces BERT performance by less than 10%. Even though BERT was never exposed to randomly shuffled text during pretraining, it easily adapts to this setting, which suggests that BERT is largely performing lexical reasoning over each (context, answer) pair.

Finally, when the context is removed and the words in each ending are shuffled, performance drops to 60.4%. While low, this is still higher than ELMo's performance (<60% from Zellers et al., 2018). As neither context nor structure is needed to discriminate between human and machine-written endings in a majority of cases, it is likely that systems primarily learn to detect distributional stylistic patterns during finetuning.

3.3 Where do the stylistic biases come from?

SWAG was constructed via Adversarial Filtering (AF). Endings were generated via a language model, and then selected to fool a discriminator. To understand why it was solved requires understanding the interplay of AF with respect to SWAG's generators and discriminators.

Zellers et al. (2018) used a two-layer LSTM for generation, with shallow stylistic adversarial filters.[6] This setup was robust against ELMo models, but has the shallow LM in particular produced distributional artifacts that BERT picks up on?

To investigate this, we perform AF using BERT-Large as the discriminator[7] in two settings, comparing generations from Zellers et al. (2018) with those from a finetuned GPT (Radford et al., 2018).

Strikingly, the results, Figure 5 (left), show that the generations used in SWAG are so different from the human-written endings that *AF never drops the accuracy to chance*; instead, it converges to roughly 75%. On the other hand, GPT's generations are good enough that BERT accuracy drops below 30% over many random subsplits of the data, revealing the importance of the generator.

4 *HellaSwag*

The success of BERT implies that high-quality generators and discriminators are crucial to AF's success. However, it does *not* imply that the underlying task of commonsense NLI – as opposed to a single dataset – is solved. To evaluate this claim requires us to try making a new evolution of the SWAG dataset, one in which artifacts are removed. In this section, we do just that by introducing *HellaSwag*.

SWAG는 ActivityNet, LSMDC 데이터셋을 모두 포함시키지만, HellaSWAG는 ActivityNet 데이터셋만 사용

4.1 ActivityNet Captions

We start by including video captions from the ActivityNet Captions dataset (Krishna et al., 2017). The original SWAG dataset contains these, along with captions from LSMDC (Rohrbach et al., 2017), but for *HellaSwag* we solely used

[6]The discriminator was an ensemble that featured a bag of words model, a shallow CNN, a multilayer perceptron operating on language model perplexities.

[7]On each iteration, BERT-Large is re-initialized from its pretrained checkpoint, finetuned, and then evaluated in a four-way setting on the dummy test set of held-out data. See Supp A for a details of our BERT-Large AF setup.

ActivityNet. In addition to temporal descriptions, ActivityNet also provides activity labels for each caption (e.g. `jumping rope`). We will use these activity labels as additional structure to test generalization ability.

4.2 WikiHow: A New Testbed

We next consider a new and challenging testbed for commonsense reasoning: completing how-to articles from WikiHow, an online how-to manual. We scrape 80k context and follow-up paragraphs from WikiHow, covering such diverse topics as "how to make an origami owl" to "how to survive a bank robbery." Each context has at most three sentences, as do the follow-ups.

AF's effectiveness in this new setting is shown in Figure 5 (right). We consider three settings, corresponding to endings that are either one, two, or three sentences long. In all cases, BERT performance begins high (70-90%), but there are enough generations for Adversarial Filtering to lower the final accuracy considerably. While the one-sentence case converges to slightly higher than random – 35% when it converges – the two and three sentence cases are higher, at 40% and 50% respectively. Given more context, it becomes easier to classify an ending as machine- or human-written. We compromise and use two-sentence generations. Particularly in the two-sentence case, we find ourselves in a Goldilocks zone wherein generations are challenging for deep models, yet as we shall soon see, easy for humans.

4.3 Obtaining high human agreement

How well can humans distinguish human-written endings from machine generations refined with Adversarial Filtering? In Figure 6, we compare human performance with that of BERT on a random 80%/20% split. We see a contrast between the ActivityNet and WikiHow performance. While ActivityNet starts off harder for BERT (25.5%), it also proves difficult for humans (60%). In contrast, WikiHow starts easier for BERT (41.1%) and humans find the domain almost trivial (93.5%). We hypothesis this discrepancy is due to the lengths of both datasets (Figure 7). WikiHow's 2-sentence generations average 41 tokens, versus 13 for ActivityNet. This gives WikiHow generations three times as many opportunities to make a detectable mistake.

To ensure high agreement on ActivityNet, we perform several rounds of human filtering, increasing human performance to 94%. During human validation, crowd workers are given a context and six ending choices, of which one is the true ending, and the other five are from AF. On each iteration, we replace machine-written endings that the worker rated as realistic with new samples. In the end, we keep the 25k best ActivityNet contexts (i.e. those with highest agreement among workers [8]) and the 45k best WikiHow contexts.

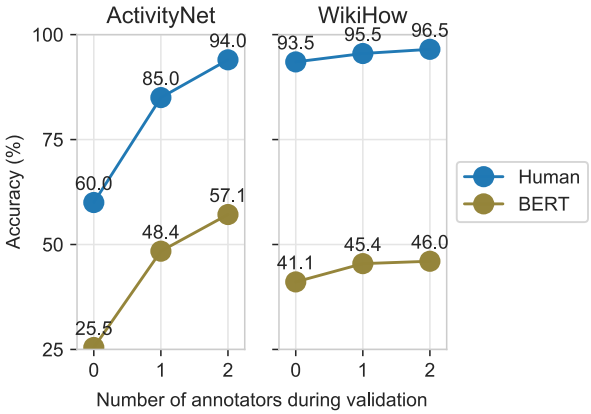

Figure 6: For *HellaSwag*, we ensure high human agreement through several rounds of annotation. By collecting how likely each ending is we can filter false negative endings – machine generations that sound realistic – and replace them with true negatives. On both sub-datasets, BERT performance increases during validation, but the gap to human performance remains wide.

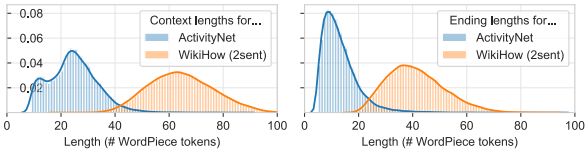

Figure 7: Lengths of ActivityNet and WikiHow; the latter with two-sentence generations. WikiHow is much longer, which corresponds to being easier for humans, while taking longer for AF to converge.

4.4 Zero-shot categories for evaluation

To evaluate a model's ability to generalize to new situations, we use category labels from WikiHow and ActivityNet to make 'zero-shot' evaluation sets. For each set (validation or test), we craft two subsets: one containing 5k 'in-domain' examples that come from categories as seen during training (Figure 8), and another with 5k 'zero-shot' examples from randomly chosen held-out categories. In total, there are 70k dataset examples.

[8] See the appendix for details about how we estimate this.

Model	Overall		In-Domain		Zero-Shot		ActivityNet		WikiHow	
	Val	Test	Val	Test	Val	Test	Val	Test	Val	Test
Split Size→	10K	10K	5K	5K	5K	5K	3.2K	3.5K	6.8K	6.5K
Chance					25.0					
fastText	30.9	31.6	33.8	32.9	28.0	30.2	27.7	28.4	32.4	33.3
LSTM+GloVe	31.9	31.7	34.3	32.9	29.5	30.4	34.3	33.8	30.7	30.5
LSTM+ELMo	31.7	31.4	33.2	32.8	30.4	30.0	33.8	33.3	30.8	30.4
LSTM+BERT-Base	35.9	36.2	38.7	38.2	33.2	34.1	40.5	40.5	33.7	33.8
ESIM+ELMo	33.6	33.3	35.7	34.2	31.5	32.3	37.7	36.6	31.6	31.5
OpenAI GPT	41.9	41.7	45.3	44.0	38.6	39.3	46.4	43.8	39.8	40.5
BERT-Base	39.5	40.5	42.9	42.8	36.1	38.3	48.9	45.7	34.9	37.7
BERT-Large	**46.7**	**47.3**	**50.2**	**49.7**	**43.3**	**45.0**	**54.7**	**51.7**	**42.9**	**45.0**
Human	95.7	95.6	95.6	95.6	95.8	95.7	94.0	94.0	96.5	96.5

Table 1: Performance of models, evaluated with accuracy (%).We report results on the full validation and test sets (Overall), as well as results on informative subsets of the data: evaluated on in-domain, versus zero-shot situations, along with performance on the underlying data sources (ActivityNet versus WikiHow). All models substantially underperform humans: the gap is over 45% on in-domain categories, and 50% on zero-shot categories.

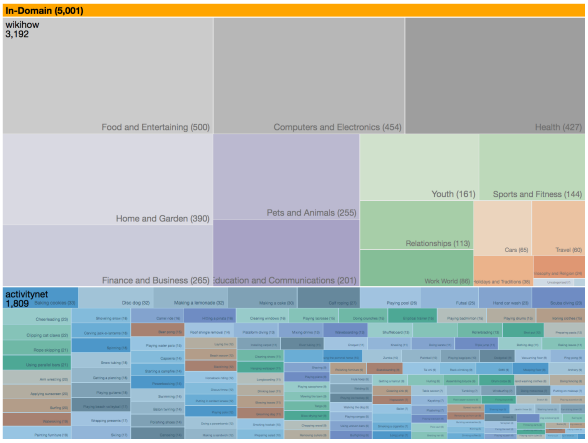

Figure 8: Examples on the in-domain validation set of *HellaSwag*, grouped by category label. Our evaluation setup equally weights performance on categories seen during training as well as out-of-domain.

5 Results

We evaluate the difficulty of *HellaSwag* using a variety of strong baselines, with and without massive pretraining. The models share the same format: given a context and an ending, return a *logit* for that ending. Accordingly, we train our models using a four-way cross-entropy loss, where the objective is to predict the correct ending. In addition to BERT-Large, our comparisons include:

a. **OpenAI GPT** (Radford et al., 2018): A fine-tuned 12-layer transformer that was pre-trained on the BookCorpus (Zhu et al., 2015).

b. **Bert-Base**: A smaller version of the BERT model whose architecture size matches GPT.

c. **ESIM+ELMo** (Chen et al., 2017; Peters et al., 2018): This is the best-performing ELMo model for NLI, modified slightly so the final output layer is now a four-way softmax over endings.

d. **LSTM sentence encoder**: This is a randomly initialized two-layer bi-LSTM; the second layer's hidden states are max-pooled and fed into an MLP to predict the logit. We consider three variations: GloVe embeddings, ELMo embeddings, or (frozen) BERT-Base embeddings.[9]

e. **FastText**: (Joulin et al., 2017) An off-the-shelf library for bag-of-words text classification.[10]

We compare all models to human performance by asking five independent crowd workers to solve the same four-way multiple choice problems; their predictions are combined via majority vote.

Our results, shown in Table 1, hint at the difficulty of the dataset: human performance is over 95%, while overall model performance is below 50% for every model. Surprisingly, despite BERT-Large having been used as the adversarial filter, it still performs the strongest at 47.3% overall. By making the dataset adversarial for BERT, it seems to also have become adversarial **for every other model.** For instance, while ESIM+ELMo obtained 59% accuracy on SWAG, it obtains only 33.3% accuracy on *HellaSwag*.

In addition to pretraining being critical, so too is end-to-end finetuning. Freezing BERT-Base and adding an LSTM on top lowers its overall performance 4.3%. This may help explain why models such as ESIM+ELMo struggled on SWAG, as ELMo isn't updated during finetuning.

While BERT is the best model, it still struggles on *HellaSwag*, and especially so on zero-shot cat-

[9]For ELMo and BERT-Base, the model learns scalar weights to combine each internal layer of the encoder.

[10]This model is trained with binary cross entropy loss.

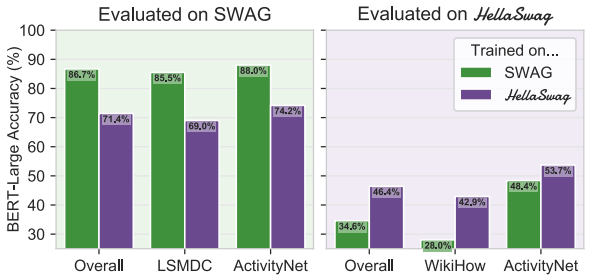

Figure 9: Transfer experiments from SWAG to *HellaSwag* and vice versa, evaluated on the validation sets. Overall, a BERT-Large that is trained on SWAG hardly generalizes to *HellaSwag*: it scores 34.6%.

egories. Performance drops roughly 5% on the test fold, which suggests that the finetuning is not enough for BERT to learn to generalize to novel activities or how-to categories.

Last, we see that WikiHow is a much harder domain that ActivityNet for machines: 45% Bert-Large performance, versus 96.5% for humans. Curiously, it is on this source dataset that we see the smallest gap between OpenAI GPT and BERT. In fact, OpenAI GPT outperforms BERT on WikiHow, but the reverse is true for ActivityNet. One possibility is that the left-to-right structure of GPT is the right inductive bias for WikiHow - perhaps reasoning bidirectionally over long contexts is too much for a 12-layer transformer to learn.

5.1 SWAG to *HellaSwag* transfer

Given the shared goals and partial domains of SWAG and *HellaSwag*, it is natural to ask to what extent models can transfer between the two datasets. In Figure 9 we show the results from transfer experiments: models are trained on one dataset and evaluated on the other.[11]

The best models are trained on the same dataset that they are evaluated on: training on SWAG and evaluating on *HellaSwag* lowers performance by 12%; vice versa lowers performance by 15%. The missing domain for *HellaSwag* models is movie descriptions (LSMDC), still, *HellaSwag* models obtain 69% accuracy. On the other hand, SWAG models do not generalize at all to their missing domain, WikiHow (28%), suggesting that learning general commonsense reasoning was hardly necessary to solve SWAG.

[11]Note that the ActivityNet splits are different for each dataset. To avoid skewing the results, we report only on the validation video captions that are not in the training sets of either dataset. The overall accuracy is then a weighted average, where ActivityNet examples are weighted proportionately more. This gives a slight advantage to training on SWAG, as it sees all the ActivityNet categories when training.

Category: Shaving (ActivityNet; In-domain)

A bearded man is seen speaking to the camera and making several faces. the man

a) then switches off and shows himself via the washer and dryer rolling down a towel and scrubbing the floor. (0.0%)
b) then rubs and wipes down an individual's face and leads into another man playing another person's flute. (0.0%)
c) is then seen eating food on a ladder while still speaking. (0.0%)
d) then holds up a razor and begins shaving his face. (100.0%)

Category: Sharpening knives (ActivityNet; Zero-Shot)

Two men are in a room and the man with a blue shirt takes out a bench stone and with a little lubricant on the stone takes an knife and explains how to sharpen it. then he

a) uses a sharpener to smooth out the stone using the knife. (100.0%)
b) shows how to cut the bottom with the knife and place a tube on the inner and corner. (0.0%)
c) bends down and grabs the knife and remove the appliance. (0.0%)
d) stops sharpening the knife and takes out some pieces of paper to show how sharp the knife is as he cuts slivers of paper with the knife. (0.0%)

Category: Youth (WikiHow; In-Domain)

How to make up a good excuse for your homework not being finished

Blame technology. One of the easiest and most believable excuses is simply blaming technology. You can say your computer crashed, your printer broke, your internet was down, or any number of problems.

a) Your excuses will hardly seem believable. [substeps] This doesn't mean you are lying, just only that you don't have all the details of how your computer ran at the time of the accident. (0.0%)
b) The simplest one to have in a classroom is to blame you entire classroom, not just lab. If you can think of yourself as the victim, why not blame it on technology. (9.4%)
c) Most people, your teacher included, have experienced setbacks due to technological problems. [substeps] This is a great excuse if you had a paper you needed to type and print. (29.1%)
d) It may also be more believable if you are fully aware that you may be flying at high speed on a plane and need someone to give you traffic report. Your problem might be your laptop failing to charge after a long flight. (61.5%)

Figure 10: Example questions answered by BERT-Large. Correct model predictions are **blue**, incorrect predictions are red. The right answers are **bolded**.

5.2 Qualitative examples

We show several qualitative examples in Figure 10, along with BERT-Large's predictions. BERT does well on some ActivityNet contexts, such as in the first row, where it correctly predicts the ending for a `shaving` caption. Whereas *shaving* is in-domain, the second example about `sharpening knives` is zero-shot. In this context, BERT's answer suggests that one would use a knife to sharpen a stone, rather than vice versa. The last example comes from WikiHow, which appears to be incredibly challenging for BERT. BERT picks answer **d**, which has more words that match the context of *technology* (planes, traffic, laptop), but is incoherent.[12]

[12]Among other issues, why would someone suddenly be aware that they are 'flying at high speed on a plane...?'

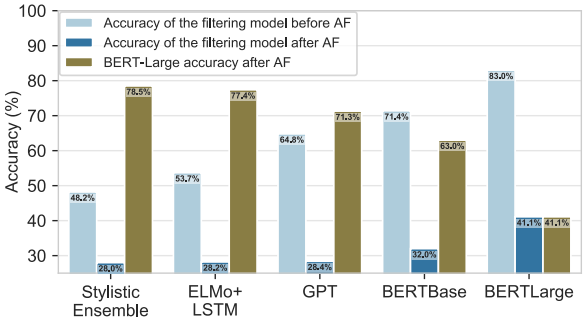

Figure 11: Performance on the WikiHow subset of alternative variations of *HellaSwag*, where different Adversarial Filters are used (but without human validation). We consider the shallow stylistic adversaries used by Zellers et al. (2018) (Stylistic Ensemble), as well as an LSTM with ELMo embeddings, GPT, BERT-Base, and BERT-Large. For each adversarial filtering model, we record the accuracy of that model before and after AF is used. We also evaluate each alternative dataset using BERT-Large. The results suggest that using a a stronger model at test time (over the model used for AF) improves performance, but is not enough to solve the task.

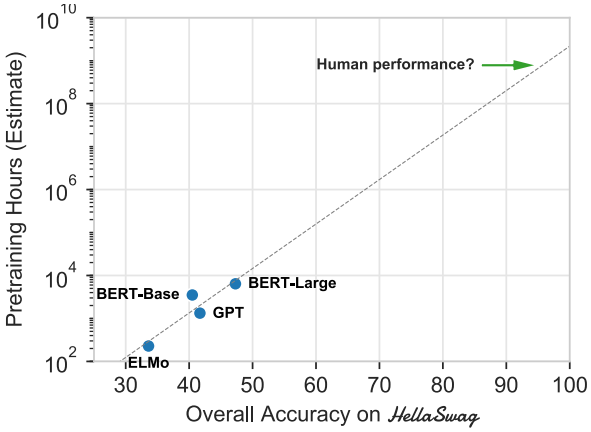

Figure 12: Estimated pretraining hours required to reach a desired accuracy on *HellaSwag*. We estimate perfomance with respect to a RTX 2080 Ti - a modern, fast GPU, and fit a log-linear regression line. An extrapolation suggests that to reach human-level performance on *HellaSwag*, without algorithmic or computational improvements, would require 10^9 GPU-hours of pretraining (over 100k GPU years).

6 Discussion

Our results suggest that *HellaSwag* is a challenging testbed for state-of-the-art NLI models, even those built on extensive pretraining. The question still remains, though, of *where will the field go next?*

6.1 How easy might *HellaSwag* be for future discriminators?

In this paper, we showed the existence of a Goldilocks zone of text complexity – in which generations are nonsensical, but existing state-of-the-art NLP models cannot tell the difference. How hard will the dataset be for future, even more powerful, models?

Answering this question is challenging because *these models don't exist (or are unavailable) at the time of writing*. However, one remedy is to perform an ablation study on the Adversarial Filtering model used, comparing weaker filters with stronger discriminators. We present our results in Figure 11, and find that while weak discriminators (like the stylistic ensemble used to make SWAG) only marginally reduce the accuracy of BERT-Large, increasing the gap between the filter and the final discriminator is not enough to solve the task. For instance, using a discriminator with 3x the parameters as the adversarial filter (BERT-Large vs. BERT-Base) results in 63% machine accuracy.

6.2 How well does pretraining scale?

Overall, the current paradigm of pretraining large models on lots of data has made immense progress on NLP benchmarks. Though we expect this trend to continue, it also behooves us to consider its limits. If more compute is indeed the answer for human-level commonsense inference, what would the compute requirements of this hypothetical massive model look like?

We investigate this in Figure 12 by comparing the accuracies of known models on *HellaSwag* with their computational needs. This estimation is a rough estimate: we convert reported TPU runtimes to our benchmark RTX 2080 Ti GPU using the Roofline model (Williams et al., 2009), which focuses primarily on the bottleneck of loading tensors into GPU memory. Extrapolating from an exponential fit suggests that reaching human-level performance on our dataset would require 10^9 GPU hours, or 100k years – unless algorithmic improvements are made.

What might these algorithmic improvements look like? These could include architectural advances, better pretraining objectives, and beyond. However, these improvements share the bottleneck of the data source. To answer some *HellaSwag* questions correctly without reasoning deeply – like knowing that it is a bad idea to stop at a red light for 'at most two seconds' – might require an exponential number of samples, due to prob-

lems of reporting bias (Gordon and Van Durme, 2013). Alternatively, future models might answer correctly only by picking up on spurious patterns, in which case a new development of the benchmark – using these models as adversaries – would place us in the same position as we are right now.

Put another way, for humans to answer *HellaSwag* questions requires *abstracting away* from language and modeling *world states* instead. We postulate that this is what separates solving the *task* of commonsense NLI, as opposed to a particular dataset. Indeed, we find that existing deep methods often get fooled by lexical false friends. For example, in the WikiHow example from Figure 10, BERT chooses an ending that matches the *technology* words in the context, rather than matching the deeper topic: using technology as an excuse for not doing homework.

6.3 Towards a future of evolving benchmarks

What happens when *HellaSwag* gets solved? We believe the answer is simple: crowdsource another dataset, with the same exact format, and see where models fail. Indeed, in our work we found this to be straightforward from an *algorithmic* perspective: by throwing in the *best known generator* (GPT) and the *best known discriminator* (BERT-Large), we made a dataset that is adversarial - not just to BERT, but to all models we have access to.

While this was easy algorithmically, care must be taken from a data curation standpoint. Indeed, we find success exists within a Goldilocks zone: the data source must be complex enough that state-of-the-art generators often make mistakes, while simple enough such that discriminators often fail to catch them. This ties the future of SWAG-style benchmarks to progress on language generation: until generation is solved, commonsense NLI will remain unsolved. Even recent promising results on scaling up language models (Radford et al., 2019) find problems in terms of consistency, with the best curated examples requiring 25 random seeds.

7 Conclusion

In this paper, we presented *HellaSwag*, a new dataset for physically situated commonsense reasoning. By constructing the dataset through adversarial filtering, combined with state-of-the-art models for language generation and discrimination, we produced a dataset that is adversarial to the most robust models available – even when models are evaluated on items from the training distribution. In turn, we provided insight into the inner workings of pretrained models, and suggest a path for NLP progress going forward: towards benchmarks that adversarially co-evolve with evolving state-of-the-art models.

Acknowledgments

We thank the reviewers, as well as Jesse Thomason, for their helpful feedback. We thank the Mechanical Turk workers for their great work during dataset collection. Thanks also to Zak Stone and the Google Cloud TPU team for help with the computing infrastructure. This work was supported by the National Science Foundation through a Graduate Research Fellowship (DGE-1256082) and NSF grants (IIS-1524371, 1637479, 165205, 1703166), the DARPA CwC program through ARO (W911NF-15-1-0543), the IARPA DIVA program through D17PC00343, the Sloan Research Foundation through a Sloan Fellowship, the Allen Institute for Artificial Intelligence, the NVIDIA Artificial Intelligence Lab, and gifts by Google and Facebook. The views and conclusions contained herein are those of the authors and should not be interpreted as representing endorsements of IARPA, DOI/IBC, or the U.S. Government.

References

Yonatan Belinkov and Yonatan Bisk. 2018. Synthetic and natural noise both break neural machine translation. In *ICLR*. ICLR.

Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, Si Wei, Hui Jiang, and Diana Inkpen. 2017. Enhanced lstm for natural language inference. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1657–1668.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Max Glockner, Vered Shwartz, and Yoav Goldberg. 2018. Breaking nli systems with sentences that require simple lexical inferences. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 650–655.

Jonathan Gordon and Benjamin Van Durme. 2013. Reporting bias and knowledge acquisition. In *Proceed-*

ings of the 2013 workshop on Automated knowledge base construction, pages 25–30. ACM.

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel R. Bowman, and Noah A. Smith. 2018. Annotation artifacts in natural language inference data. In *Proc. of NAACL*.

Ari Holtzman, Jan Buys, Maxwell Forbes, and Yejin Choi. 2019. The curious case of neural text degeneration. *arXiv preprint arXiv:1904.09751*.

Robin Jia and Percy Liang. 2017. Adversarial examples for evaluating reading comprehension systems. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2021–2031.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2017. Bag of tricks for efficient text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, volume 2, pages 427–431.

Ranjay Krishna, Kenji Hata, Frederic Ren, Li Fei-Fei, and Juan Carlos Niebles. 2017. Dense-Captioning Events in Videos. In *International Conference on Computer Vision (ICCV)*.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, volume 1, pages 2227–2237.

Adam Poliak, Jason Naradowsky, Aparajita Haldar, Rachel Rudinger, and Benjamin Van Durme. 2018. Hypothesis only baselines in natural language inference. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics*, pages 180–191.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. Technical report, OpenAI.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. Technical report, OpenAI.

Anna Rohrbach, Atousa Torabi, Marcus Rohrbach, Niket Tandon, Christopher Pal, Hugo Larochelle, Aaron Courville, and Bernt Schiele. 2017. Movie Description. *International Journal of Computer Vision*, 123(1):94–120.

Rachel Rudinger, Vera Demberg, Ashutosh Modi, Benjamin Van Durme, and Manfred Pinkal. 2015. Learning to predict script events from domain-specific text. In *Proceedings of the Fourth Joint Conference on Lexical and Computational Semantics*, pages 205–210.

Samuel Williams, Andrew Waterman, and David Patterson. 2009. Roofline: An insightful visual performance model for floating-point programs and multicore architectures. Technical report, Lawrence Berkeley National Lab.(LBNL), Berkeley, CA (United States).

Rowan Zellers, Yonatan Bisk, Roy Schwartz, and Yejin Choi. 2018. Swag: A large-scale adversarial dataset for grounded commonsense inference. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Yukun Zhu, Ryan Kiros, Richard Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *arXiv preprint arXiv:1506.06724*.

Supplemental Material

A Adversarial Filtering Setup

In this subsection, we provide some more details regarding the Adversarial Filtering experiments.

Our version of Adversarial Filtering is mostly the same as Zellers et al. (2018). Details:

a. On each iteration, we split the dataset up into 80% training and 20% testing. We don't do anything special for this split (like looking at the video/article IDs).

b. For ActivityNet, we use $k = 9$ assigned indices for every example. (This corresponds to the number of red columns in Figure 2). For WikiHow, we used $k = 5$, since we found that there were fewer good endings produced by the generators after scaling up the sequence length.

c. Similarly to Zellers et al. (2018), we train the AF models in a multi-way fashion. Since we use BERT-Large as the discriminator, this matches Devlin et al. (2018)'s model for SWAG: on each training example, the model is given exactly one positive ending and several negative endings, and the model computes probability distribution over the endings through a softmax. However, we also wanted to always report 4-way probability for simplicity. To do this, we train in a 4-way setting (the training set is constructed by subsampling 3 wrong answers from the set of k that are currently assigned to each example). The accuracy values that are reported are done so using the first 3 assigned negatives in dataset $\mathcal{D}_{test}$.

d. Sometimes, BERT never converges (accuracy around 25%), so when this happens, we don't do the reassignment.

B GPT Setup

We generate our dataset examples from OpenAI GPT. We finetune the model for two epochs on WikiHow, and 5 epochs on ActivityNet, using the default learning rate of (Radford et al., 2018). Importantly, we generate randomly according to the language model distribution, rather than performing beam search – this would bias the generations towards common words. For the WikiHow endings, we used Nucleus Sampling with $p = 0.98$, which means that the probability weights for the tail (those tokens with cumulative probability mass < 0.02) are zeroed out (Holtzman et al., 2019).

C BERT setup

We extensively study BERT in this paper, and make no changes to the underlying architecture or pretraining. For all of the experiments where we provide context, we set up the input to the BERT model like this:

`[CLS] A woman is outside with a bucket and a dog. The dog is running around trying to avoid a bath. [SEP] She gets the dog wet, then it runs away again [SEP]`

In the case where only the ending is provided, we adopt the BERT-style 'single-span' setting: `[CLS] She gets the dog wet, then it runs away again [SEP]`

D A discussion on BERT Hyperparameters and Instability

It is worth noting that many of our experiments some instability. On the SWAG experiments, we use the same hyperparameters as (Devlin et al., 2018) - these generally work very well.[13] However, we find that they become a bit unstable when crossing over to make *HellaSwag*. Here, we discuss some strategies and insight that we picked up on.

a. We use a batch size of 64 examples rather than 16, and warm the model up for 20% of the dataset (rather than 10%). This helps the model adapt to SWAG more gradually, without diverging early on.

b. For the Adversarial Filtering experiments (for both WikiHow and ActivityNet), we randomize some of the hyperaparmeters on each iteration. We sample a learning rate between `1e-5` and `4e-5`, using a log-uniform distribution. These outer ranges were recommended from the original BERT paper. Additionally, with probability 0.5 we use the cased model (where the input isn't originally lowercased before tokenization), rather than the uncased model.

c. During adversarial filtering, we used 3 epochs. However, we found that adding more epochs

[13] The only exception is for the plots where we vary the number of training examples. In this case, we don't want to disadvantage the trials without much training data (since this would allow for fewer parameter updates). To remedy this, we continue training for 10 epochs and report the best validation performance over the entire training history.

helped the model during fine-tuning on the final dataset *HellaSwag*. Our best configuration uses 10 epochs.

d. While fine-tuning on *HellaSwag* we used a learning rate of `2e-5`.

E Human validation

We performed human validation using the same setup as (Zellers et al., 2018). Humans get six answers to choose from, of which exactly one is the true ending and the other five are from AF. We found that multiple rounds of human validation were especially helpful on ActivityNet. However, it helps to do the human validation in an intelligent way: if the first worker is confused, the answer should be replaced before it goes to the next worker. This is a hard problem, so we adopt the following approach:

a. We use best practices on mechanical turk, paying workers fairly (up to 37 cents per HIT on WikiHow). We also used a qualification HIT that was autograded to help filter for workers who are good at the task. Workers who tended to prefer the generated endings over the real ones were dequalified from participating.

b. For each worker, we use the summary of their performance so far to estimate $P(\text{answer } i \text{ is right} | \text{worker rates } i \text{ as best})$. We can then use this to estimate how confident we are in each answer choice: we want to be confident that workers will *not* prefer the wrong answers. Also, this allows us to aggregate performance across crowd workers, by multiplying the probabilities for each answer choice.

c. On each round of filtering, we *keep* the 3 wrong endings that workers least prefer (based on the probability scores, along with the right ending. The other two endings are new ones.

Particularly on ActivityNet, we found that there are some contexts where the ground truth answer isn't liked by workers. To fix this, we end up taking the best 25k examples from ActivityNet and the best 45k from WikiHow. (By best, we mean the ones with the highest probability that workers will predict the true answer, versus the three easiest-to-guess negatives, as judged by the Naive Bayes model). We make Figure 7 ('The road to *HellaSwag*') by doing this process (taking the best examples) for each dataset, while varying the number of annotators that are used for getting the scores for each ending. (In the case where there are 0 annotators, we get a random sample).

F Human Evaluation

We do a human evaluation while giving workers the exact same task as is given to the models. Workers are given five endings, and must pick the best one. We obtain human evaluation numbers by combining 5 turkers together, with a majority vote.

We found that the biggest differences in difficulty in humans were due to domain (WikiHow is easier than ActivityNet). To account for this, we did the human evaluation over 200 examples from WikiHow, and 200 examples from ActivityNet, for each number of previous validators as shown in Figure 7 (0, 1, or 2). To report the accuracy of a split that's mixed between WikiHow and ActivityNet, we use the following formula:

$$\frac{acc_{WikiHow} \cdot N_{WikiHow} + acc_{ActivityNet} \cdot N_{ActivityNet}}{N_{WikiHow} + N_{ActivityNet}}$$

Here, *acc* refers to the accuracy on each dataset as judged by humans, and N is the number of examples from that dataset in the split.

G More examples

We additionally have more validation examples, shown in Figure 2.

H In-Domain and Zero-Shot categories

See Figure 13 for a closer look at the dataset categories.

Category: Preparing pasta (activitynet; indomain)

A kitchen is shown followed by various ingredients and a woman speaking to the camera. She begins showing the ingredients and putting them into a hot boiling pot and stirring around. she

a) shows off the oven and begins assembling the cookies in the oven by pushing a button on the oven. (2.2%)
b) continues mixing up more ingredients and then puts them all together in a bowl, serving the dish ad sprinkling olive oil around it. (97.8%)
c) shows raising and lowering the pot until adding more water and corn syrup. (0.0%)
d) places an omelette onto the screen and puts it in the oven to bake. (0.0%)

Category: Doing crunches (activitynet; indomain)

We see a fitness center sign. We then see a man talking to the camera and sitting and laying on a exercise ball. the man

a) demonstrates how to increase efficient exercise work by running up and down balls. (0.0%)
b) moves all his arms and legs and builds up a lot of muscle. (80.9%)
c) then plays the ball and we see a graphics and hedge trimming demonstration. (0.0%)
d) performs sits ups while on the ball and talking. (19.1%)

Category: Sharpening knives (activitynet; zeroshot)

A man is seen spinning a blade with his foot on a machine and moving his hands up with down holding a knife. the camera

a) pans around and shows a woman moving around in a jump rope machine. (0.0%)
b) captures him from several angles while he sharpens the knife with complete concentration. (81.6%)
c) pans around and points to a man standing inside the machine as the man continues to move on the machine. (18.4%)
d) then pans around to a woman and her daughter who also dance at the show. (0.0%)

Category: Layup drill in basketball (activitynet; zeroshot)

A female basketball coach is seen instructing a group of girl basketball players who are standing in line on a basketball court. the first girl

a) passes to another coach and then runs to the net and takes a layup. (0.0%)
b) trying to get the ball to go far past the basket and hit it back towards the basket while her coach continues teaching her. (100.0%)
c) walks across the court with the ball and keeps walking then pulling the girls to the other side of the court and the girls begin playing volleyball rhythmically rolling on the floor as the coach helps them follow how to properly do things. (0.0%)
d) line up and stand behind a dummy dummy. (0.0%)

Category: Youth (wikihow; indomain)

[header] How to make up a good excuse for your homework not being finished [title] Blame technology. [step] One of the easiest and most believable excuses is simply blaming technology. You can say your computer crashed, your printer broke, your internet was down, or any number of problems.

a) Your excuses will hardly seem believable. [substeps] This doesn't mean you are lying, just only that you don't have all the details of how your computer ran at the time of the accident. (0.0%)
b) The simplest one to have in a classroom is to blame you entire classroom, not just lab. If you can think of yourself as the victim, why not blame it on technology. (9.4%)
c) Most people, your teacher included, have experienced setbacks due to technological problems. [substeps] This is a great excuse if you had a paper you needed to type and print. (29.1%)
d) It may also be more believable if you are fully aware that you may be flying at high speed on a plane and need someone to give you traffic report. Your problem might be your laptop failing to charge after a long flight. (61.5%)

Category: Family Life (wikihow; zeroshot)

[header] How to raise your children to be helpers [title] Call them helpers when you ask for things. [step] Instead of asking for help, ask your child to " be a helper. " all people, children included, are more motivated when their identity is in play.

a) You can start doing this with your children as early as two years old. [substeps] You might say, " jayden, can you be a helper and clean your bedroom before grandma comes over? " or " please be a helper and stay quiet while your sister naps. (0.1%)
b) When you call your child helpers, describe what they do and what they need to be helped for. [substeps] You could say, " i need you to help dad during his lunch break at work. (99.9%)
c) If you ask your child for things they have access to, it encourages them to put more effort into making things happen. [substeps] To make sure they understand exactly what's expected of them, you could try saying, " i'm looking for helpers who can be helpers. (0.0%)
d) Call them when you need them for help or for monetary help. [substeps] For example, if you need help with something you don't know how to do, let your child know you're excited to help with this. (0.0%)

Table 2: Example questions answered by BERT-Large. Correct model predictions are in **blue**, incorrect model predictions are red. The right answers are **bolded**.

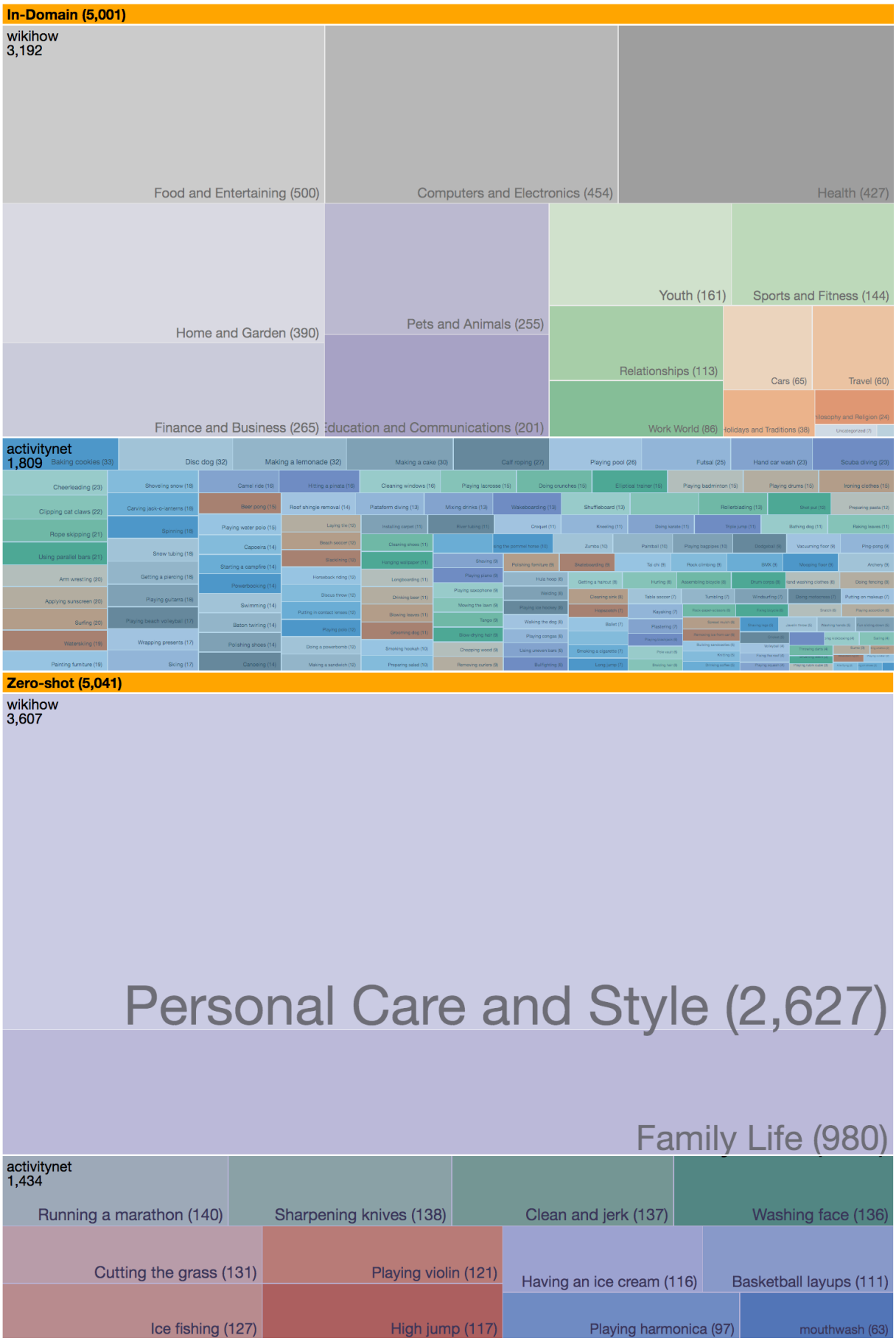

Figure 13: Examples on the in-domain validation set of *HellaSwag*, grouped by category label. Our evaluation setup equally weights performance on categories seen during training as well as out-of-domain.