FinerWeb-10BT: Refining Web Data with LLM-Based Line-Level Filtering

Erik Henriksson*
University of Turku
erik.henriksson@utu.fi

Otto Tarkka*
University of Turku
ohitar@utu.fi

Filip Ginter
University of Turku
figint@utu.fi

Abstract

Data quality is crucial for training Large Language Models (LLMs). Traditional heuristic filters often miss low-quality text or mistakenly remove valuable content. In this paper, we introduce an LLM-based line-level filtering method to enhance training data quality. We use GPT-4o mini to label a 20,000-document sample from FineWeb at the line level, allowing the model to create descriptive labels for low-quality lines. These labels are grouped into nine main categories, and we train a DeBERTa-v3 classifier to scale the filtering to a 10B-token subset of FineWeb. To test the impact of our filtering, we train GPT-2 models on both the original and the filtered datasets. The results show that models trained on the filtered data achieve higher accuracy on the HellaSwag benchmark and reach their performance targets faster, even with up to 25% less data. This demonstrates that LLM-based line-level filtering can significantly improve data quality and training efficiency for LLMs. We release our quality-annotated dataset, FinerWeb-10BT, and the codebase to support further work in this area.

1 Introduction

In recent years, the size of large language models (LLMs) and their training datasets has expanded tremendously, as companies and researchers strive to build increasingly capable models. In fact, if current trends continue, we may run out of human-generated text data within a decade (Villalobos et al., 2024). This has led to a growing interest in data quality over quantity: rather than only expanding datasets, researchers are exploring ways to achieve high performance with smaller, cleaner datasets. Recent studies suggest that removing low-quality text from training data can improve model performance, even when the overall size of the dataset is reduced (Longpre et al., 2023).

Furthermore, training state-of-the-art (SOTA) language models requires significant computational resources, which are expensive and, depending on the power source, can contribute to climate change. For example, the carbon emissions from training GPT-3 have been estimated at 552 tCO_2e (Patterson et al., 2021), while Meta reports that training the 405 billion parameter Llama 3.1 emitted 8,930 tCO_2e (Meta-Llama, 2024). Smaller, but higher quality datasets will speed up training and, thus, high-quality data are necessary to train not only better models but also greener ones.

While several publicly available datasets are used for training LLMs, many recent datasets are still cleaned using simple heuristic filters, which often leave substantial amounts of low-quality text while potentially discarding clean text. Machine-learning techniques offer a promising alternative, as they enable models to identify patterns related to data quality. However, labeling data to train such models is a tedious and time-consuming process. In this paper, we address these issues by investigating the following research questions (RQs):

RQ1: How well can an LLM identify low-quality content missed by heuristic filters?

RQ2: Does LLM-based quality filtering of training datasets improve model performance?

To examine these questions, we analyze FineWeb, a dataset that claims to provide "the finest text data at scale" (Penedo et al., 2024). Using GPT-4o mini (OpenAI, 2024a), we label a 20,000-document sample from FineWeb, classifying each line as either *Clean* or belonging to one

*These authors contributed equally.

Proceedings of the Joint 25th Nordic Conference on Computational Linguistics and 11th Baltic Conference on Human Language Technologies (NoDaLiDa/Baltic-HLT 2025), pages 258–268
March 3-4, 2025

of several low-quality categories, such as *copyright notice*, *programming code*, or *formatting elements*. Instead of defining a label taxonomy ourselves, we allow the model to generate its own labels as needed, resulting in 547 unique low-quality labels. After refining these labels, we group them into nine broader categories for easier classification. Next, we train a DeBERTa-v3 (He et al., 2021) classifier using the labeled data to scale the filtering process. This classifier allows us to automatically detect low-quality content in a larger 10B-token sample of FineWeb. Finally, we evaluate the impact of LLM-based filtering by training GPT-2 models (Radford et al., 2019) on both the filtered and unfiltered datasets.

We release our quality-annotated dataset, FinerWeb-10BT, available at https://huggingface.co/datasets/TurkuNLP/finerweb-10bt. The code to replicate our experiments is also provided at https://github.com/TurkuNLP/finerweb-10bt.

2 Background

A recent survey by Albalak et al. (2024) discusses the many steps involved in selecting data for training LLMs, including language filtering, deduplication, removal of toxic or explicit content, and heuristic-based data quality filtering. Our focus here is on the latter two—data filtering and heuristic approaches—using an LLM-driven approach to refine data quality more precisely. As Albalak et al. (2024) note, there is no universal standard for "high-quality" data. In this work, we define it as human-written, continuous English text from the main content of a website, reflecting natural language use across diverse contexts and domains. Examples include core text from interviews, forum posts, news articles, blogs, and recipes. In contrast, low-quality content includes recurring elements like navigational menus, copyright notices, programming code, and metadata.

Given that LLMs require vast amounts of text data for training, the Internet has become a primary source for these data. Since 2008, CommonCrawl has collected a corpus of approximately 10 petabytes of web content (Baack, 2024). Despite its size, CommonCrawl is neither a complete nor fully representative sample of the Internet, but it serves as a foundational source for building refined datasets used in LLM training. Here, we focus on three major datasets sourced from CommonCrawl: C4 (Raffel et al., 2023), RefinedWeb (Penedo et al., 2023), and FineWeb. These datasets use different preprocessing techniques to filter out unwanted material, each with its strengths and weaknesses. We discuss these datasets because their preprocessing methods are well-documented, which allows us to make meaningful comparisons.

All three datasets extract plaintext from HTML documents. C4 uses the WET files provided by CommonCrawl, which come with pre-extracted plaintext, whereas RefinedWeb and FineWeb use trafilatura[1] to extract text directly from HTML. Although trafilatura and similar tools remove much of the unwanted noise, further preprocessing is often required. For instance, Penedo et al. (2023) note that "many documents remain interlaced with undesirable lines" despite using trafilatura. Deduplication and language filtering are also important aspects of document cleaning but we do not focus on them in this paper, as they are specialized techniques not directly related to line-level text quality.

Existing filtering methods can be grouped into three levels based on their precision: document level, line level, and character level. By far the most common method is document level filtering, which removes entire documents based on simple rules. Examples include filtering documents with phrases like "lorem ipsum", documents with fewer than three sentences, or documents with excessive repetition. Line level filtering targets specific lines within documents, removing lines that contain terms like "javascript", consist solely of numbers, or fall below a certain length threshold. Character level filtering is less common and is only applied in one of the three datasets: in C4, citation markers commonly found in Wikipedia, such as "[1]" and "[citation needed]", are removed.

Document level heuristic filtering is efficient for quickly removing large volumes of low-quality data, but it can result in the loss of substantial high-quality text. In contrast, line and character level filtering provide more precision by targeting specific content but they require significantly more computational resources at scale. Simple heuristics, such as removing lines that contain the word "javascript" can be hit or miss, sometimes discarding valuable data along with the low-quality content. Given the vast size of datasets like Com-

[1] https://trafilatura.readthedocs.io/en/latest/

monCrawl, creating a simple filtering system that only removes undesirable content without impacting valuable data is nearly impossible. The filters that are used are also often dataset and language specific. For example, FineWeb applies a heuristic that removes documents where "the fraction of lines shorter than 30 characters is >= 0.67" (Penedo et al., 2024, p. 7), but this threshold was determined through extensive manual testing and is specific to that dataset.

An ideal quality filter would work across languages and datasets, avoiding trial-and-error by focusing on actual text quality rather than proxies like line length or keywords. It should also be efficient, removing only low-quality content while keeping valuable data intact. LLMs bring us closer to this goal: rather than using heuristics, they assess text quality directly, enabling granular filtering, even within mostly clean documents. Since LLMs are effective at producing fluent and readable text, they are likely well suited to identifying high-quality text across different languages and datasets. However, it should be noted that while SOTA LLMs are fluent in English and other high-resource languages, their performance in low-resource languages is consistently worse (Li et al., 2024). In this study, we only analyze English documents, and care should be taken before generalizing the results to other languages or multilingual datasets.

The use of LLMs for quality filtering is a relatively new approach, and best practices are still emerging. For instance, Dubey et al. (2024) utilize Llama 2 to assess the quality of web documents for training Llama 3, but details of their methodology are vague. The recent trend of withholding full training datasets for SOTA models has made it difficult to understand the extent to which LLMs are currently used in data preprocessing (Nguyen et al., 2024; Maini et al., 2024). Other efforts, such as those by Wettig et al. (2024), involve ranking documents based on quality using GPT-3.5, evaluating factors such as style, educational value, and factuality. Similarly, Llama 3 was used to create the FineWedEdu dataset by evaluating educational content quality, and Gunasekar et al. (2023) employ GPT-4 to annotate code datasets based on educational value.

Our approach differs from prior work by focusing on general-purpose data quality improvements rather than curating specialized datasets. We aim to broadly enhance training data quality through LLM-driven filtering that removes low-quality lines with minimal manual intervention. This allows us to assess how automated filtering can improve training data and, ultimately, model performance in foundation model training.

3 Methods

Our data source is FineWeb (Penedo et al., 2024), a 15-trillion-token collection of English text sourced from CommonCrawl and preprocessed with standard heuristics. The preprocessing includes steps such as length thresholds, string matching, language and URL filtering, and deduplication. Despite these measures, the authors of FineWeb acknowledge that the dataset could benefit from further refinement. For more details on the preprocessing steps, see the original paper (Penedo et al., 2024). In our study, we use a 10B-token (1̃5 million documents) sample from FineWeb, FineWeb-10BT[2].

Our preprocessing pipeline consists of several steps. First, we use GPT-4o mini (OpenAI, 2024a) to label a sample of 20,000 documents from FineWeb at the line level. The model is tasked with generating descriptive labels for each line, categorizing them as either high-quality (*Clean*) or into low-quality categories. This labeling process is data-driven, allowing the model to create a dynamic labeling scheme rather than relying on predefined categories. Previous research has shown that LLMs can be used to annotate data and create label taxonomies (Wan et al., 2024).

Next, we use OpenAI's o1-preview model (OpenAI, 2024c) to group the numerous labels generated by GPT-4o mini into a smaller, more manageable set. This forms the basis of a classification system, which we use to train a small encoder-based classifier. This classifier scales the labeling process by assigning quality scores throughout the FineWeb-10BT dataset, enabling line-level filtering of low-quality content.

To evaluate our filtering, we train GPT-2 models (Radford et al., 2019) on both the cleaned and original versions of FineWeb-10BT. We compare model performances using the HellaSwag benchmark (Zellers et al., 2019), a widely used test for commonsense reasoning in language models. This allows us to assess whether the filtering improves

[2]https://huggingface.co/datasets/HuggingFaceFW/fineweb/viewer/sample-10BT

training data quality and model performance.

Given the complexity of Internet text data (Laippala et al., 2023), defining low-quality categories in advance is challenging. Our data-driven approach, by contrast, allows the LLM to dynamically create labels based on the content it encounters, rather than relying on fixed categories. We believe this approach enables a more flexible and detailed analysis of low-quality content in FineWeb compared to rule-based methods or predefined categorizations.

4 Experiments and results

4.1 Labeling FineWeb using GPT-4o mini

We begin by labeling a 20,000-document sample from FineWeb-10BT using the GPT-4o mini model. The model is prompted to classify each line as either *Clean* (high quality and suitable for training large language models) or assign a descriptive label if the line contains low-quality content, such as HTML tags or random symbols. Initially, the model generates its own descriptive labels, which are then added to a list for subsequent classification. As the model processes more documents, it selects labels from the existing list or creates new ones if necessary. To avoid bias from label order, the list is shuffled after each iteration.

We split the documents into batches of up to 15 consecutive lines. The model receives a prompt, a list of labels, and a batch of lines. Since the lines are consecutive, each one is evaluated in context, providing the model with more information for accurate labeling. For documents containing a single line longer than 200 characters, the line is split into segments of no more than 200 characters, using sentence-ending punctuation as the split point. This prevents output errors, which we observed when processing excessively long lines during preliminary tests. Segmenting these lines also enables more precise analysis.

This process results in quality labels for 328,472 lines. Of these, 274,343 lines (83%) are labeled as *Clean*. For low-quality lines, the model generates 547 unique descriptive labels. However, we find that many of these labels are assigned to one line only; in fact, 142 labels appear only once. Upon inspection, we notice many of the lines could be considered high-quality and, thus, to streamline the label set, we map all these infrequent labels to *Clean*. For the remaining labels, we take a sample of lines and manually verify that they represent genuinely low-quality content. If the majority of lines for a particular label are of high quality, we remap that label to *Clean*. After this refinement, the number of descriptive labels is reduced to 382, with 45,205 lines (14%) classified as low-quality. Conversely, 86% of the dataset is now labeled *Clean*.

To visualize the distribution of these classes, we generate a 2D UMAP projection (McInnes et al., 2018) of the 50 most frequent label embeddings, created using the Stella-en-400M-v5 model (StellaEncoder, 2024) (see also Section 4.3 below). The UMAP projection reduces the original 1024-dimensional embeddings to 2D, as shown in Figure 1, with each dot scaled to represent the relative frequency of each class.

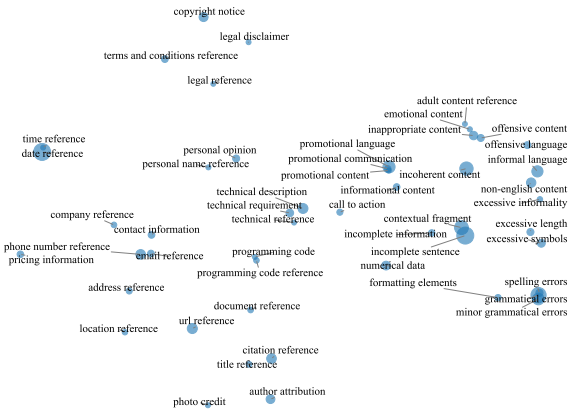

Figure 1: UMAP plot of embeddings of the 50 most frequent LLM-generated label names, created using the Stella-en-400M-v5 model.

Inspecting the plot, we observe that certain types of low-quality content tend to occupy distinct regions in the space. For instance, legal texts appear in the top-left, adult and toxic content in the top center-right, and bibliographic references near the bottom. Contact information, such as times, dates, and phone numbers, is loosely grouped on the left, while technical content, like programming code, appears in the center. These patterns suggest that the LLM-generated labels capture meaningful line quality distinctions and form a useful basis for our final class set.

4.2 Grouping the labels

The next step in our pipeline is to group the 382 detailed labels into a more concise set of broader, more manageable categories, which simplifies training the encoder classifier. We use Ope-

nAI's o1-preview, a newly released "reasoning" model (OpenAI, 2024b), to organize the labels. We instruct the model to create clear, distinct categories that assign each label to only one group. The goal is to produce a set of classes that the classifier can learn and differentiate easily.

Category	Lines	%
Clean	283,267	86.24
Formatting, Style & Errors	13,150	4.00
Bibliographical & Citation References	8,768	2.67
Promotional & Spam Content	7,339	2.23
Contact & Identification Information	3,898	1.19
Navigation & Interface Elements	3,327	1.01
Technical Specifications & Metadata	3,298	1.00
Legal & Administrative Content	2,992	0.91
Offensive or Inappropriate Content	2,433	0.74
Total	**328,472**	**100**

Table 1: Label categories and the number of lines in each category.

After manually inspecting the output, we find that the groupings are mostly accurate, though some manual corrections are necessary. For example, the model occasionally fails to assign all labels or places some labels into multiple categories. After fixing these issues, we finalize a classification scheme with 9 broader categories, as shown in Table 1.

To verify that the labels match human intuition, we conduct a manual inter-annotator agreement (IAA) evaluation on a random sample of 50 documents (726 lines). Two human annotators, familiar with the 9-label class set, assess whether they agree or disagree with the LLM-generated labels. In cases of disagreement, they provide corrected labels. We compute Cohen's Kappa scores comparing human ratings with the LLM's for both the full label set and a simplified binary classification (*Clean* vs. *Non-clean*).

	A1	A2	Avg.
All labels	0.79	0.60	0.70
Clean vs. Non-clean	0.78	0.67	0.73

Table 2: Cohen's Kappa scores for human annotators (A1 and A2) vs. the GPT-4o mini generated labels (LLM).

As shown in Table 2, Cohen's Kappa for the full label set is 0.788 for Annotator 1 (A1) and 0.604 for Annotator 2 (A2), with an average of 0.70, indicating moderate to substantial agreement. For the binary classification, Kappa scores improve slightly, with A1 at 0.78 and A2 at 0.67, averaging 0.73. This suggests that while agreement varies, the LLM-based classification generally produces acceptable labels for the FineWeb texts.

These results address RQ1, which examines how well an LLM can identify low-quality content that heuristic filters miss. The LLM's classifications align well with those of human annotators, showing that it succees to detect low-quality lines overlooked by earlier heuristic methods applied to FineWeb data. While there is some variability in the IAA scores, the overall performance supports our LLM-driven approach.

4.3 Training a classifier

To scale our labeling process for the FineWeb-10BT dataset, we use encoder-based models, which are faster, more cost-effective, and often better suited to classification than large generative LLMs. We experiment with four models: DeBERTa-v3 (base and large variants) (He et al., 2021), Stella-en-400M-v5 (currently the top model of its size for English text clustering on the MTEB leaderboard (Muennighoff et al., 2023)[3]), and XLM-RoBERTa-base (Conneau et al., 2019). The first three models are English-only, while XLM-RoBERTa is multilingual.

For line-by-line classification, we first extract individual lines from the documents, treating each as a separate example. The data is then shuffled and split into training (70%), development (10%), and test (20%) sets using stratification. We add a classification head to each model to generate probabilities across the 9 classes for each line and fine-tune both the classification head and base model. Preliminary tests showed that this approach yielded better results than training only the classification head with a frozen base model.

For training, we use bfloat16 precision, a learning rate of 1e-5, and a batch size of 16. Early stopping is applied with a patience of 5 based on evaluation loss, with a maximum of 5 epochs; however, models typically converge after the first epoch. We also apply label smoothing (0.1) to the cross-entropy loss to improve generalization. Training is done on a single A100 GPU.

[3]https://huggingface.co/spaces/mteb/leaderboard

	μ F1	M F1	Clean		
			P	R	F1
DeBERTa-v3-base	0.81	0.66	**0.88**	0.91	**0.90**
DeBERTa-v3-large	0.81	0.65	0.87	0.92	0.89
Stella-en-400M-v5	0.81	**0.67**	0.87	0.92	0.89
XLM-RoBERTa-base	0.80	0.63	0.86	0.92	0.89

Table 3: Comparison of Classifiers on Multiclass Classification using the held-out test set. μ F1: Micro F1, M F1: Macro F1, P: Precision, R: Recall, F1: F1 score for the *Clean* class.

Table 3 presents the evaluation results of the models on the test set. We report micro and macro F1 scores for all classes, along with precision, recall, and F1 for the *Clean* class. The results show that the models perform similarly, with micro F1 scores ranging between 0.80 and 0.81, and macro F1 scores between 0.63 and 0.67. For the *Clean* class, precision ranges from 0.86 to 0.88, recall from 0.91 to 0.92, and F1 between 0.89 and 0.90. These metrics indicate strong performance in distinguishing between high- and low-quality content, though the lower macro F1 score suggests some classes are less easily distinguishable. Additionally, newer or larger models do not significantly improve performance. Thus, for subsequent analyses, we select the DeBERTa-v3-base model.

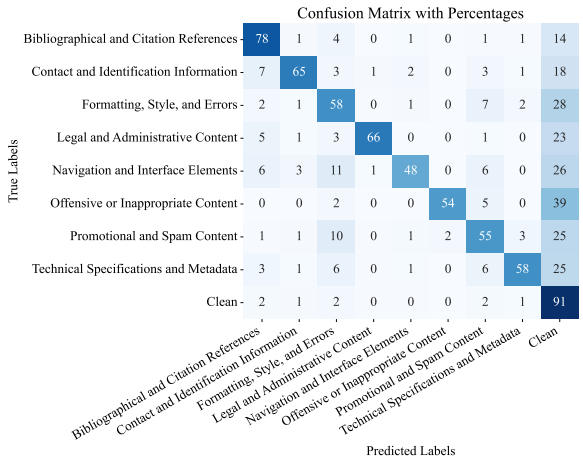

Figure 2: Confusion matrix of predictions from our line quality classifier on the test set.

To further examine the performance of the classifier and spot common misclassifications, we evaluate its predictions on the held-out test set using DeBERTa-v3-base and display the results in a confusion matrix (Figure 2). Most misclassifications fall into the *Clean* class, indicating strong separation between the other classes. The least distinct class is *Offensive or Inappropriate Content*, likely due to the inherent difficulty in defining clear boundaries for offensive material in LLM training datasets. In contrast, *Bibliographical and Citation References* stands out as the most distinct class, likely due to its easily recognizable formatting and content.

We note that it is preferable for the classifier to err on the side of labeling low-quality lines *Clean* (as shown in the confusion matrix and evaluation scores) rather than mistakenly tagging high-quality lines as low-quality. This bias helps reduce the risk of discarding valuable data from the dataset.

4.4 Cleaning FineWeb

Given our classifier's promising evaluation results, we now label the 10B-token subset of FineWeb using our DeBERTa-v3-base classifier. For this task, we simplify to binary classification by focusing only on the probability of the *Clean* class versus all other classes combined, where probabilities closer to 1 indicate high-quality content.

Although the classifier performs well, the *Clean* class makes up 86% of the data, which may cause the model to produce overconfident predictions for this class. To correct for this imbalance, we apply Platt scaling (Platt et al., 1999) to adjust the predicted probabilities, aiming for a more accurate reflection of the true probability distribution and more reliable thresholding. Specifically, we train a Platt logistic regression model on the held-out test set and apply it on top of the classifier when predicting quality scores for the FineWeb-10BT dataset.

We predict the quality labels for the FineWeb-10BT dataset in shards of 100,000 documents. Within each shard, we process batches of 128 lines, grouping lines by length to speed up processing. We then add a "quality_score" key to each document, with each item scored from 0 to 1 to four decimal places.

Figure 3 shows a histogram of the quality scores for a 1-million-line sample from FineWeb-10BT, with calibrated probabilities binned in 10% intervals on a logarithmic scale. The distribution is bimodal, with most lines receiving high-quality scores. About 75% of lines score above 0.90, while 8% score below 0.50. Most of the data is concentrated in the highest quality bin (90–100%), with a smaller cluster confidently assigned very

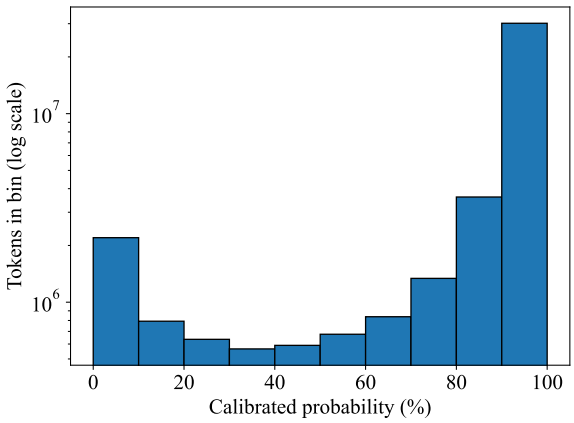

Figure 3: Quality probabilities for a 1M-line sample from FineWeb-10BT, binned in 10% intervals (log scale). A total of 8% of lines fall below the 0.50 quality threshold, and 25% fall below the 0.90 threshold.

low scores, indicating that the classifier effectively separates high-quality from low-quality lines.

Table 4 shows examples of lines with the highest and lowest quality scores according to our classifier. The highest-scoring lines are coherent, context-rich sentences, while the lowest-scoring lines contain metadata, copyright symbols, tags, and formatting artifacts, demonstrating that the method performs as intended.

4.5 Evaluation with GPT-2 and HellaSwag

Finally, we evaluate our data cleaning process by pre-training small GPT-2 models (124M parameters) on three versions of the dataset: (1) the original 10B-token sample from FineWeb, (2) a filtered version with a 0.50 quality score threshold, reducing the dataset by 8%, and (3) a version with a 0.90 quality score threshold, reducing data by 25%. The training code is adapted from Khajavi (2024), with modifications specific to our experimental setup.

The models are trained for 18,994 steps (a single epoch on the full FineWeb-10BT dataset) using four A100 GPUs. Every 200 steps, we evaluate model performance on the HellaSwag benchmark (Zellers et al., 2019), which is widely used to assess the ability of language models to complete sentences in commonsense reasoning contexts. To account for inherent randomness, we repeat the training on all datasets five times each, with each run lasting approximately 5 hours and 30 minutes.

Figure 4 shows the evaluation results, which

Line	Score
Lines with highest quality scores	
She hopes taking part in the 5K will encourage others to become or stay active.	0.9674
I'd love it if you'd visit and give me your impressions and/or suggestions.	0.9659
We aim to make the ceremony an enjoyable celebration.	0.9657
prayerfully seek peace for our partners in Nigeria.	0.9655
I loved the way this shirt looked and thought it would be cool to wear it.	0.9655
Lines with lowest quality scores	
|Also published as|US20040168193|	0.0057
|Tags:||Anglesey, Beach, General, Landscape, Landscape / travel, Lighthouse, Llanddwyn, Sea, Sunrise, Wales, Water|	0.0056
|FOR IMMEDIATE RELEASE||PRESS RELEASE #MR12-003881|	0.0055
|©Sunwest Bank||||Equal Housing Lender||||Member FDIC|	0.0051
- ©- copyright & copy; or & #169; or & #xA9;	0.0050

Table 4: Examples of highest and lowest quality lines from a 1M-line FineWeb-10BT sample, with their probabilities of being *Clean*.

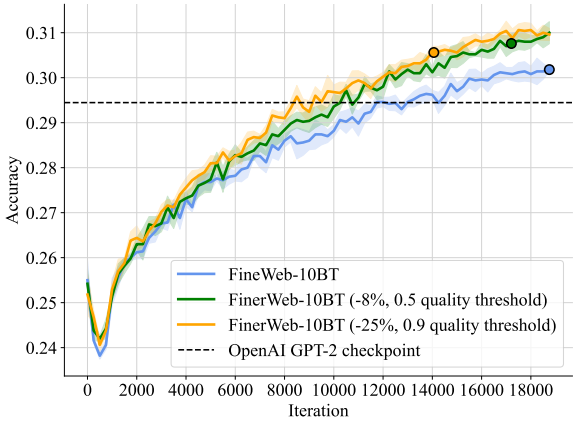

Figure 4: Average HellaSwag accuracy over 5 runs for three models: the original FineWeb-10BT and two cleaned versions with quality thresholds of 0.50 (8% data reduction) and 0.90 (25% data reduction). Dot markers indicate epoch ends for each dataset run. GPT-2 (124M) checkpoint accuracy is shown for reference.

indicate a clear positive impact from our data cleaning process. Models trained on the cleaner FinerWeb-10BT datasets—both the 8% and 25% reduced versions—consistently outperform those

trained on the original FineWeb-10BT data. By the end of 18,994 training steps, both cleaned versions show an average HellaSwag evaluation score that is 0.1 points higher than that of the original dataset. This improvement is robust, as shown by the shaded areas around the lines, representing standard deviations that suggest the effect is unlikely due to random variation across runs.

Additionally, both cleaned models achieve slightly higher HellaSwag accuracy than the original FineWeb-10BT model at their respective epoch ends, as indicated by the colored dots in the plot. Remarkably, both models reach the original dataset's highest score approximately 6k steps earlier, a 32% reduction in training time. This means a reduction of roughly 1 hour and 45 minutes, based on our 5 hour 30 minute run time per training round. Interestingly, the 25% reduced dataset shows a slight edge over the 8% cleaned data, although the difference is minimal; both clean models ultimately reach an average HellaSwag score of 0.31 within the same number of steps. This suggests that a more aggressive data cleaning strategy could be worth exploring in future work. In summary, our data cleaning process produces models that (1) reach target accuracy faster and (2) achieve higher accuracy within the same training time, addressing our RQ2.

5 Discussion

The labels generated by GPT-4o mini reveal both the quantity and types of low-quality lines that remain in FineWeb. The largest categories include lines with grammatical errors, poor formatting, and incomplete sentences, along with recurring items like time stamps, legal jargon, and promotional content. While these elements do not necessarily reduce dataset quality (a good language model should recognize items like copyright notices or phone numbers), our evaluation shows that reducing their prevalence improves both accuracy and training efficiency. These findings suggest that more precise control over the types and proportions of low-quality data included could further benefit model performance. Even when simplified to binary classification, our LLM-driven approach clearly outperforms heuristic methods in enhancing dataset quality.

Specifically, our evaluation on GPT-2 using HellaSwag shows that with less but cleaner data, the model achieves comparable or even slightly better accuracy. While GPT-2 is small relative to SOTA models, our results provide strong evidence that LLM-based data filtering can reduce training time and save energy. Although we tested our method on a small, English-only dataset, this data-driven approach to quality filtering is easily adaptable to other datasets and languages, although low-resource language may suffer from worse LLM performance.

Using an LLM as a judge of text quality introduces some bias, as the model's training data and design choices influence the resulting labels. For example, mature SOTA LLMs have strong in-built safety features that prevent them from generating harmful or offensive content. In our case, we observe that GPT-4o mini sometimes labels mild expletives, such as "shut up", as toxic, reflecting an overly sensitive filter for offensive language. As described in Sections 4.1 and 4.2 we made some manual adjustments to the LLM labeling to account for such biases. Also, the line between low-quality and high-quality is naturally vague, which introduces noise into the data. In future work, we plan to experiment with different models and adjust our prompts to further improve this filtering approach.

6 Conclusion

In this paper, we propose a novel approach to improving the quality of large-scale language model training datasets through fine-grained, line-level filtering with large language models (LLMs). We first used GPT-4o mini to label a sample from the FineWeb dataset, generating detailed labels that captured low-quality content often overlooked by heuristic filters, addressing our first research question (RQ1). These labels were grouped into broader categories using OpenAI's o1-preview model, followed by training a DeBERTa-v3 classifier to scale the filtering across FineWeb-10BT. Our experiments demonstrate that this LLM-driven filtering pipeline improves model performance (addressing RQ2), as GPT-2 models trained on the filtered dataset achieved higher HellaSwag accuracy with up to 25% less data than those trained on the original FineWeb-10BT dataset.

These findings suggest that traditional heuristic filters may not be sufficient and that more sophisticated data preprocessing methods are necessary, especially as we face challenges like data scarcity and environmental concerns. Our approach con-

tributes to the emerging field of LLM-based data preprocessing, offering a promising avenue for improving training efficiency and model performance.

In future work, we plan to refine our pipeline by broadening the labeling scheme to provide a more comprehensive description of document contents. We will also experiment with more nuanced filtering approaches, moving beyond simple score-based thresholds, and compare against baselines such as random data reduction to further validate our filtering method. We also plan to test Llama-style models and other architectures to see how our findings scale to newer LLMs. Further evaluations and statistical testing will help strengthen the reliability of our results. Finally, we plan to extend our method to other datasets and languages.

Acknowledgments

This project has received funding from the European Union's Horizon Europe research and innovation programme under Grant agreement No 101070350 and from UK Research and Innovation (UKRI) under the UK government's Horizon Europe funding guarantee [grant number 10052546]. The contents of this publication are the sole responsibility of its authors and do not necessarily reflect the opinion of the European Union.

This work was supported by the Research Council of Finland.

Computational resources for this study were provided by CSC — IT Center for Science.

References

Alon Albalak, Yanai Elazar, Sang Michael Xie, Shayne Longpre, Nathan Lambert, Xinyi Wang, Niklas Muennighoff, Bairu Hou, Liangming Pan, Haewon Jeong, Colin Raffel, Shiyu Chang, Tatsunori Hashimoto, and William Yang Wang. 2024. A survey on data selection for language models.

Stefan Baack. 2024. A critical analysis of the largest source for generative ai training data: Common crawl. In *The 2024 ACM Conference on Fairness, Accountability, and Transparency (FAccT'24)*.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. *CoRR*, abs/1911.02116.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurelien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Roziere, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Graeme Nail, Gregoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel Kloumann, Ishan Misra, Ivan Evtimov, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, Khalid El-Arini, Krithika Iyer, Kshitiz Malik, Kuenley Chiu, Kunal Bhalla, Lauren Rantala-Yeary, Laurens van der Maaten, Lawrence Chen, Liang Tan, Liz Jenkins, Louis Martin, Lovish Madaan, Lubo Malo, Lukas Blecher, Lukas Landzaat, Luke de Oliveira, Madeline Muzzi, Mahesh Pasupuleti, Mannat Singh, Manohar Paluri, Marcin Kardas, Mathew Oldham, Mathieu Rita, Maya Pavlova, Melanie Kambadur, Mike Lewis, Min Si, Mitesh Kumar Singh, Mona Hassan, Naman Goyal, Narjes Torabi, Nikolay Bashlykov, Nikolay Bogoychev, Niladri Chatterji, Olivier Duchenne, Onur Çelebi, Patrick Alrassy, Pengchuan Zhang, Pengwei Li, Petar Vasic, Peter Weng, Prajjwal Bhargava, Pratik Dubal, Praveen Krishnan, Punit Singh Koura, Puxin Xu, Qing He, Qingxiao Dong, Ragavan Srinivasan, Raj Ganapathy, Ramon Calderer, Ricardo Silveira Cabral, Robert Stojnic, Roberta Raileanu, Rohit Girdhar, Rohit Patel, Romain Sauvestre, Ronnie Polidoro, Roshan Sumbaly, Ross Taylor, Ruan Silva, Rui Hou, Rui Wang, Saghar Hosseini, Sahana Chennabasappa, Sanjay Singh, Sean Bell, Seohyun Sonia Kim, Sergey Edunov, Shaoliang Nie, Sharan Narang, Sharath Raparthy, Sheng Shen, Shengye Wan, Shruti Bhosale, Shun Zhang, Simon Vandenhende, Soumya Batra, Spencer Whitman, Sten Sootla, Stephane Collot, Suchin Gururangan, Sydney Borodinsky, Tamar Herman, Tara Fowler, Tarek Sheasha, Thomas Georgiou, Thomas Scialom, Tobias Speckbacher, Todor Mihaylov, Tong Xiao, Ujjwal Karn, Vedanuj Goswami, Vibhor Gupta, Vignesh Ramanathan, Viktor Kerkez, Vincent Gonguet,

Virginie Do, Vish Vogeti, Vladan Petrovic, Weiwei Chu, Wenhan Xiong, Wenyin Fu, Whitney Meers, Xavier Martinet, Xiaodong Wang, Xiaoqing Ellen Tan, Xinfeng Xie, Xuchao Jia, Xuewei Wang, Yaelle Goldschlag, Yashesh Gaur, Yasmine Babaei, Yi Wen, Yiwen Song, Yuchen Zhang, Yue Li, Yuning Mao, Zacharie Delpierre Coudert, Zheng Yan, Zhengxing Chen, Zoe Papakipos, Aaditya Singh, Aaron Grattafiori, Abha Jain, Adam Kelsey, Adam Shajnfeld, Adithya Gangidi, Adolfo Victoria, Ahuva Goldstand, Ajay Menon, Ajay Sharma, Alex Boesenberg, Alex Vaughan, Alexei Baevski, Allie Feinstein, Amanda Kallet, Amit Sangani, Anam Yunus, Andrei Lupu, Andres Alvarado, Andrew Caples, Andrew Gu, Andrew Ho, Andrew Poulton, Andrew Ryan, Ankit Ramchandani, Annie Franco, Aparajita Saraf, Arkabandhu Chowdhury, Ashley Gabriel, Ashwin Bharambe, Assaf Eisenman, Azadeh Yazdan, Beau James, Ben Maurer, Benjamin Leonhardi, Bernie Huang, Beth Loyd, Beto De Paola, Bhargavi Paranjape, Bing Liu, Bo Wu, Boyu Ni, Braden Hancock, Bram Wasti, Brandon Spence, Brani Stojkovic, Brian Gamido, Britt Montalvo, Carl Parker, Carly Burton, Catalina Mejia, Changhan Wang, Changkyu Kim, Chao Zhou, Chester Hu, Ching-Hsiang Chu, Chris Cai, Chris Tindal, Christoph Feichtenhofer, Damon Civin, Dana Beaty, Daniel Kreymer, Daniel Li, Danny Wyatt, David Adkins, David Xu, Davide Testuggine, Delia David, Devi Parikh, Diana Liskovich, Didem Foss, Dingkang Wang, Duc Le, Dustin Holland, Edward Dowling, Eissa Jamil, Elaine Montgomery, Eleonora Presani, Emily Hahn, Emily Wood, Erik Brinkman, Esteban Arcaute, Evan Dunbar, Evan Smothers, Fei Sun, Felix Kreuk, Feng Tian, Firat Ozgenel, Francesco Caggioni, Francisco Guzmán, Frank Kanayet, Frank Seide, Gabriela Medina Florez, Gabriella Schwarz, Gada Badeer, Georgia Swee, Gil Halpern, Govind Thattai, Grant Herman, Grigory Sizov, Guangyi, Zhang, Guna Lakshminarayanan, Hamid Shojanazeri, Han Zou, Hannah Wang, Hanwen Zha, Haroun Habeeb, Harrison Rudolph, Helen Suk, Henry Aspegren, Hunter Goldman, Ibrahim Damlaj, Igor Molybog, Igor Tufanov, Irina-Elena Veliche, Itai Gat, Jake Weissman, James Geboski, James Kohli, Japhet Asher, Jean-Baptiste Gaya, Jeff Marcus, Jeff Tang, Jennifer Chan, Jenny Zhen, Jeremy Reizenstein, Jeremy Teboul, Jessica Zhong, Jian Jin, Jingyi Yang, Joe Cummings, Jon Carvill, Jon Shepard, Jonathan McPhie, Jonathan Torres, Josh Ginsburg, Junjie Wang, Kai Wu, Kam Hou U, Karan Saxena, Karthik Prasad, Kartikay Khandelwal, Katayoun Zand, Kathy Matosich, Kaushik Veeraraghavan, Kelly Michelena, Keqian Li, Kun Huang, Kunal Chawla, Kushal Lakhotia, Kyle Huang, Lailin Chen, Lakshya Garg, Lavender A, Leandro Silva, Lee Bell, Lei Zhang, Liangpeng Guo, Licheng Yu, Liron Moshkovich, Luca Wehrstedt, Madian Khabsa, Manav Avalani, Manish Bhatt, Maria Tsimpoukelli, Martynas Mankus, Matan Hasson, Matthew Lennie, Matthias Reso, Maxim Groshev, Maxim Naumov, Maya Lathi, Meghan Keneally, Michael L. Seltzer, Michal Valko, Michelle Restrepo, Mihir Patel, Mik Vyatskov, Mikayel Samvelyan, Mike Clark, Mike Macey, Mike Wang, Miquel Jubert Hermoso, Mo Metanat, Mohammad Rastegari, Munish Bansal, Nandhini Santhanam, Natascha Parks, Natasha White, Navyata Bawa, Nayan Singhal, Nick Egebo, Nicolas Usunier, Nikolay Pavlovich Laptev, Ning Dong, Ning Zhang, Norman Cheng, Oleg Chernoguz, Olivia Hart, Omkar Salpekar, Ozlem Kalinli, Parkin Kent, Parth Parekh, Paul Saab, Pavan Balaji, Pedro Rittner, Philip Bontrager, Pierre Roux, Piotr Dollar, Polina Zvyagina, Prashant Ratanchandani, Pritish Yuvraj, Qian Liang, Rachad Alao, Rachel Rodriguez, Rafi Ayub, Raghotham Murthy, Raghu Nayani, Rahul Mitra, Raymond Li, Rebekkah Hogan, Robin Battey, Rocky Wang, Rohan Maheswari, Russ Howes, Ruty Rinott, Sai Jayesh Bondu, Samyak Datta, Sara Chugh, Sara Hunt, Sargun Dhillon, Sasha Sidorov, Satadru Pan, Saurabh Verma, Seiji Yamamoto, Sharadh Ramaswamy, Shaun Lindsay, Shaun Lindsay, Sheng Feng, Shenghao Lin, Shengxin Cindy Zha, Shiva Shankar, Shuqiang Zhang, Shuqiang Zhang, Sinong Wang, Sneha Agarwal, Soji Sajuyigbe, Soumith Chintala, Stephanie Max, Stephen Chen, Steve Kehoe, Steve Satterfield, Sudarshan Govindaprasad, Sumit Gupta, Sungmin Cho, Sunny Virk, Suraj Subramanian, Sy Choudhury, Sydney Goldman, Tal Remez, Tamar Glaser, Tamara Best, Thilo Kohler, Thomas Robinson, Tianhe Li, Tianjun Zhang, Tim Matthews, Timothy Chou, Tzook Shaked, Varun Vontimitta, Victoria Ajayi, Victoria Montanez, Vijai Mohan, Vinay Satish Kumar, Vishal Mangla, Vítor Albiero, Vlad Ionescu, Vlad Poenaru, Vlad Tiberiu Mihailescu, Vladimir Ivanov, Wei Li, Wenchen Wang, Wenwen Jiang, Wes Bouaziz, Will Constable, Xiaocheng Tang, Xiaofang Wang, Xiaojian Wu, Xiaolan Wang, Xide Xia, Xilun Wu, Xinbo Gao, Yanjun Chen, Ye Hu, Ye Jia, Ye Qi, Yenda Li, Yilin Zhang, Ying Zhang, Yossi Adi, Youngjin Nam, Yu, Wang, Yuchen Hao, Yundi Qian, Yuzi He, Zach Rait, Zachary DeVito, Zef Rosnbrick, Zhaoduo Wen, Zhenyu Yang, and Zhiwei Zhao. 2024. The Llama 3 Herd of Models.

Suriya Gunasekar, Yi Zhang, Jyoti Aneja, Caio César Teodoro Mendes, Allie Del Giorno, Sivakanth Gopi, Mojan Javaheripi, Piero Kauffmann, Gustavo de Rosa, Olli Saarikivi, Adil Salim, Shital Shah, Harkirat Singh Behl, Xin Wang, Sébastien Bubeck, Ronen Eldan, Adam Tauman Kalai, Yin Tat Lee, and Yuanzhi Li. 2023. Textbooks are all you need.

Pengcheng He, Jianfeng Gao, and Weizhu Chen. 2021. Debertav3: Improving deberta using electra-style pre-training with gradient-disentangled embedding sharing.

Matin Khajavi. 2024. https://github.com/matinkhajavi/gpt-from-scratch. `https://github.com/MatinKhajavi/GPT-from-scratch`.

Veronika Laippala, Samuel Rönnqvist, Miika Oinonen, Aki-Juhani Kyröläinen, Anna Salmela, Douglas

Biber, Jesse Egbert, and Sampo Pyysalo. 2023. Register identification from the unrestricted open web using the corpus of online registers of english. *Language Resources and Evaluation*, 57.

Zihao Li, Yucheng Shi, Zirui Liu, Fan Yang, Ali Payani, Ninghao Liu, and Mengnan Du. 2024. Language ranker: A metric for quantifying llm performance across high and low-resource languages.

Shayne Longpre, Gregory Yauney, Emily Reif, Katherine Lee, Adam Roberts, Barret Zoph, Denny Zhou, Jason Wei, Kevin Robinson, David Mimno, and Daphne Ippolito. 2023. A pretrainer's guide to training data: Measuring the effects of data age, domain coverage, quality, & toxicity.

Pratyush Maini, Skyler Seto, Richard Bai, David Grangier, Yizhe Zhang, and Navdeep Jaitly. 2024. Rephrasing the web: A recipe for compute and data-efficient language modeling. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14044–14072, Bangkok, Thailand. Association for Computational Linguistics.

Leland McInnes, John Healy, Nathaniel Saul, and Lukas Großberger. 2018. Umap: Uniform manifold approximation and projection. *Journal of Open Source Software*, 3(29):861.

Meta-Llama. 2024. Model card. Website.

Niklas Muennighoff, Nouamane Tazi, Loïc Magne, and Nils Reimers. 2023. Mteb: Massive text embedding benchmark.

Thuat Nguyen, Chien Van Nguyen, Viet Dac Lai, Hieu Man, Nghia Trung Ngo, Franck Dernoncourt, Ryan A. Rossi, and Thien Huu Nguyen. 2024. CulturaX: A cleaned, enormous, and multilingual dataset for large language models in 167 languages. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 4226–4237, Torino, Italia. ELRA and ICCL.

OpenAI. 2024a. Gpt-4o-mini.

OpenAI. 2024b. Learning to reason with llms.

OpenAI. 2024c. Openai o1-preview.

David Patterson, Joseph Gonzalez, Quoc Le, Chen Liang, Lluis-Miquel Munguia, Daniel Rothchild, David So, Maud Texier, and Jeff Dean. 2021. Carbon emissions and large neural network training.

Guilherme Penedo, Hynek Kydlíček, Loubna Ben allal, Anton Lozhkov, Margaret Mitchell, Colin Raffel, Leandro Von Werra, and Thomas Wolf. 2024. The fineweb datasets: Decanting the web for the finest text data at scale.

Guilherme Penedo, Quentin Malartic, Daniel Hesslow, Ruxandra Cojocaru, Alessandro Cappelli, Hamza Alobeidli, Baptiste Pannier, Ebtesam Almazrouei, and Julien Launay. 2023. The refinedweb dataset for falcon llm: Outperforming curated corpora with web data, and web data only.

John Platt et al. 1999. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in large margin classifiers*, 10(3):61–74.

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

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2023. Exploring the limits of transfer learning with a unified text-to-text transformer.

StellaEncoder. 2024. stella_en_400m_v5.

Pablo Villalobos, Anson Ho, Jaime Sevilla, Tamay Besiroglu, Lennart Heim, and Marius Hobbhahn. 2024. Will we run out of data? limits of llm scaling based on human-generated data.

Mengting Wan, Tara Safavi, Sujay Kumar Jauhar, Yujin Kim, Scott Counts, Jennifer Neville, Siddharth Suri, Chirag Shah, Ryen W. White, Longqi Yang, Reid Andersen, Georg Buscher, Dhruv Joshi, and Nagu Rangan. 2024. Tnt-llm: Text mining at scale with large language models. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '24, page 5836–5847, New York, NY, USA. Association for Computing Machinery.

Alexander Wettig, Aatmik Gupta, Saumya Malik, and Danqi Chen. 2024. QuRating: Selecting high-quality data for training language models.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence?