Skolkovo Institute of Science and Technology

Efficient Active Learning and Fact-Checking via Predictive Uncertainty Quantification for Natural Language Processing

Doctoral Thesis
by

Aleksandr Rubashevskii

Doctoral Program in Computational and Data Science and Engineering

Supervisor
Assistant Professor, Maxim Panov

Moscow — 2024

I hereby declare that the work presented in this thesis was carried out by myself at Skolkovo Institute of Science and Technology, Moscow, except where due acknowledgement is made, and has not been submitted for any other degree.

Candidate (Aleksandr Rubashevskii)

Supervisor (Asst. Prof. Maxim Panov)

Abstract

With the rapid development of natural language processing (NLP) in recent years, language models have been increasingly used to solve applied problems. Nevertheless, the question arises about the reliability and limitations of the models used, especially given the specifics of the particular task to be solved. In this thesis, we focus on the problem of efficiently estimating the uncertainty of language models for two tasks, namely active learning and fact-checking.

In an active learning task, the uncertainty measure helps to select the most informative data samples for labeling and subsequent model training in the presence of a large amount of unlabeled data and a limited budget. For efficient active learning, it is important to estimate for batches of data and select multiple samples for labeling at each iteration. We propose the batch active learning algorithm Large BatchBALD (LBB), which allows selecting multiple examples for labeling at each active learning step, which is computationally more feasible than alternative algorithms. Moreover, we show its experimental effectiveness by embedding the algorithms inside the ALToolbox benchmark and comparing it with state-of-the-art active learning strategies.

In a fact-checking task, one way to verify the validity of information in the absence of an external knowledge source is to estimate the uncertainty of the model with respect to the generated text. Efficient fact-checking under such conditions requires the ability to distinguish between different types of uncertainty and to evaluate only the relevant one. We present a new fact-checking method on uncertainty quantification at the token-level that removes the impact of task-irrelevant uncertainty about what claim to generate on the current step and what surface form to use. Our method Claim Conditioned Probability (CCP) measures only the uncertainty of a particular claim value expressed by the model. Experiments on the task of biography generation demonstrate strong improvements for CCP compared to the baselines for various languages and Large Language Models (LLMs).

Additionally, we present a holistic end-to-end solution for annotating the factuality of LLM-generated responses, which encompasses a multi-stage annotation scheme designed to yield detailed labels concerning the verifiability and factual inconsistencies found in LLM outputs. We further construct an open-domain document-level factuality benchmark Factcheck-Bench in three-level granularity: claim, sentence and document, aiming to facilitate the evaluation of automatic fact-checking systems.

Publications

This section presents the main publications of the author related to the PhD dissertation.

1. **Aleksandr Rubashevskii**, Daria Kotova, and Maxim Panov. Scalable batch acquisition for deep bayesian active learning. In *Proceedings of the 2023 SIAM International Conference on Data Mining (SDM)*, pages 739–747, Minneapolis, Minnesota, U.S., April 2023. Society for Industrial and Applied Mathematics. doi:10.1137/1.9781611977653.ch83. URL `https://epubs.siam.org/doi/abs/10.1137/1.9781611977653.ch83` [CORE A]

2. Akim Tsvigun, Leonid Sanochkin, Daniil Larionov, Gleb Kuzmin, Artem Vazhentsev, Ivan Lazichny, Nikita Khromov, Danil Kireev, **Aleksandr Rubashevskii**, Olga Shahmatova, Dmitry V. Dylov, Igor Galitskiy, and Artem Shelmanov. ALToolbox: A set of tools for active learning annotation of natural language texts. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 406–434, Abu Dhabi, UAE, December 2022. Association for Computational Linguistics. doi:10.18653/v1/2022.emnlp-demos.41. URL `https://aclanthology.org/2022.emnlp-demos.41` [CORE A*]

3. Ekaterina Fadeeva, **Aleksandr Rubashevskii**, Artem Shelmanov, Sergey Petrakov, Haonan Li, Hamdy Mubarak, Evgenii Tsymbalov, Gleb Kuzmin, Alexander Panchenko, Timothy Baldwin, Preslav Nakov, and Maxim Panov. Fact-checking the output of large language models via token-level uncertainty quantification. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 9367–9385, Bangkok, Thailand and virtual meeting, August 2024. Association for Computational Linguistics. doi:10.18653/v1/2024.findings-acl.558. URL `https://aclanthology.org/2024.findings-acl.558` [CORE A*]

4. Yuxia Wang, Revanth Gangi Reddy, Zain Muhammad Mujahid, Arnav Arora, **Aleksandr Rubashevskii**, Jiahui Geng, Osama Mohammed Afzal, Liangming

Pan, Nadav Borenstein, Aditya Pillai, Isabelle Augenstein, Iryna Gurevych, and Preslav Nakov. Factcheck-bench: Fine-grained evaluation benchmark for automatic fact-checkers. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 14199–14230, Miami, Florida, USA, November 2024b. Association for Computational Linguistics. doi:10.18653/v1/2024.findings-emnlp.830. URL `https://aclanthology.org/2024.findings-emnlp.830` [CORE A*]

Additional publications by the author related to uncertainty quantification during his PhD studies.

1. Vincent Plassier, Mehdi Makni, **Aleksandr Rubashevskii**, Eric Moulines, and Maxim Panov. Conformal prediction for federated uncertainty quantification under label shift. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 27907–27947. PMLR, 23–29 July 2023. URL `https://proceedings.mlr.press/v202/plassier23a.html` [CORE A*]

2. Vincent Plassier, Nikita Kotelevskii, **Aleksandr Rubashevskii**, Fedor Noskov, Maksim Velikanov, Alexander Fishkov, Samuel Horvath, Martin Takac, Eric Moulines, and Maxim Panov. Efficient conformal prediction under data heterogeneity. In *Proceedings of The 27th International Conference on Artificial Intelligence and Statistics*, volume 238 of *Proceedings of Machine Learning Research*, pages 4879–4887. PMLR, 02–04 May 2024. URL `https://proceedings.mlr.press/v238/plassier24a.html` [CORE A]

Acknowledgments

I would like to thank the people who made this thesis possible and those who helped this research journey take place. I would like to sincerely thank my supervisor, Assistant Professor Maxim Panov, for his valuable advice and assistance in both academic and life matters. His faith in me, as well as his contribution to my development, is invaluable. I want to thank my host advisor Preslav Nakov during my long-term mobility, with whom I delved into the topic of fact-checking of LLMs, and we were able to write several papers for this thesis. I would also like to thank my supervisor during my master's studies, Dmitry V. Dylov, with whom I wrote my first scientific paper and experienced what it is like to do research.

I would like to thank my talented colleagues Artem Shelmanov and Ekaterina Fadeeva, with whom I have worked closely over the past year. Also, I would like to thank our StatML lab for its research atmosphere and fun: Nikita Kotelevskii, Kirill Fedyanin, Roman Vashurin, Sasha Fishkov and Daria Kotova.

Also, I want to say "thank you" to all my math and physics teachers, for their hard work and ability to inspire: the late Valentina D. Loshkeeva, Irina V. Shindyaykina, Dmitry N. Oskorbin, Ismail M. Isaev, Konstantin V. Solomatin.

I would also like to thank my family for their support. I thank my parents, who instilled in me a passion for education and always helped me realize it. I thank my wife, who has walked this path with me with all its ups and downs, believing in me and supporting me.

Contents

List of Figures

List of Tables

Glossary

AL Active Learning. 16, 19

LLM Large Language Model. 77, 79, 80, 82, 83

NLI Natural Language Inference. 80

QA Question Answering. 83, 84

SOTA State-of-the-Art. 19

Chapter 1

Introduction

1.1 Overview

Due to the rapid development of language models and their applications, the task of uncertainty quantification becomes important and relevant. The model may have some bias as a result of training, limitations on acceptable modes of operation, or predictions and content produced by the model may be misleading and dangerous. When applying these models to problem-solving, there is a question of the criterion of how much the user can trust the predictions of the model. It is important to understand what uncertainty is contained in the language model about the problem to be solved and how it can be estimated.

One of the big topics concerning the use of uncertainty estimation in applied problems is active learning. In supervised machine learning tasks, the quality and volume of the training data play essential roles in achieving high performance. However, the process of data collection and labeling is often expensive, requiring a huge amount of time and resources [Roh et al., 2021, Paullada et al., 2021]. Therefore, Active Learning (AL) techniques, that choose the most informative samples for model training, minimizing the collection and annotation budget, are crucial in practice [Ren et al., 2021]. Active Learning methods are successfully applied to the various types of data: tabular [Tsymbalov et al., 2018], image [Gal et al., 2017], text [Shelmanov et al., 2019], audio [Riccardi and Hakkani-Tur, 2005], video [Vondrick and Ramanan, 2011] and others. Especially, AL can be helpful in the case of real-world data which requires involvement of subject-matter experts, namely medical [Budd et al., 2021] and manufacturing data [Lv et al., 2020].

We consider a pool-based active learning problem for classification tasks. It is assumed that there is a small amount of labeled data and a big unlabeled pool to select an object for annotation from. Selection procedure is carried out according to a certain criterion that is usually based on a so-called acquisition function. Acquisition function is maximized over the most informative samples in terms of model uncertainty measure or expected error. For example, as an acquisition function one uses variance reduction [Hoi et al., 2006], entropy [Sebastiani and Wynn, 2000] or mutual information (also known as BALD) [Houlsby et al., 2011] maximization and others. Then, selected samples are labeled and added to the already labeled dataset for further training.

At each step of the active learning cycle, one or several pool samples can be selected for annotation. By selecting one sample to label at each step, one can greedily assemble an optimal set of labeled data for training. However, with a large number of objects in the dataset, this strategy becomes inefficient as frequent model retraining becomes very computationally expensive. In this case, it is better to select multiple objects from the pool at each active learning step.

However, an acquisition of multiple objects at a time, leads to the problem of selecting similar training samples and data redundancy. Indeed, the majority of existing acquisition function (for example, BALD) do not take into account the interaction of samples within a batch. That's why very similar samples can be selected and the resulting active-learned training dataset can have many redundant points. It leads to model performance degradation and excessive use of resources. One of the state-of-the-art methods that partially copes with this problem is an extension of the BALD method, namely BatchBALD [Kirsch et al., 2019]. Its idea is to calculate mutual information in a batch manner using multiple network outputs as a joint random variable. This approach allows one to account for interactions between samples in a batch manner, preventing the selection of similar samples, but greatly increasing computational time and complexity. This opens up a space to explore what an active learning method can be that meets the right characteristics.

Another big applied uncertainty quantification task is a fact-checking for Large Language Models (LLMs). Large language models have become a ubiquitous and versatile tool for addressing a variety of natural language processing (NLP) tasks. People use these models for tasks including information search [Sun et al., 2023b], to ask medical questions [Thirunavukarasu et al., 2023], or to generate new content [Sun et al., 2023a]. Recently, there has been a notable shift in user behavior, indicating an increasing reliance on and trust in LLMs as primary information sources, often surpassing traditional channels. However, a significant challenge with the spread of these models is their tendency to produce "hallucinations", i.e., factually incorrect generations that contain misleading information [Bang et al., 2023, Dale et al., 2023]. This is a side-effect of the way modern LLMs are designed and trained [Kalai and Vempala, 2024].

LLM hallucinations are a major concern because the deceptive content at the surface level can be highly coherent and persuasive. Common examples include the creation of fictitious biographies or the assertion of unfounded claims. The danger is that a few occasional false claims might be easily obscured by a large number of factual statements, making it extremely hard for people to spot them. As hallucinations in LLM outputs are hard to eliminate completely, users of such systems could be informed via highlighting some potential caveats in the text, and this is where our approach can help.

Fact-checking is a research direction that addresses this problem. It is usually approached using complex systems that leverage external knowledge sources [Guo et al., 2022, Nakov et al., 2021b, Wadden et al., 2020]. This introduces problems related to the incomplete nature of such sources and notable overhead in terms of storing the knowledge. We argue that information about whether a generation is a hallucination is encapsulated in the model output itself, and can be extracted using uncertainty quantification (UQ) [Gal, 2016, Kotelevskii et al., 2022, Vazhentsev et al., 2022, 2023a]. This avoids implementing complex and expensive fact-checking systems that require additional computational overhead and rely on external resources.

Prior work has mainly focused on quantification of uncertainty for the whole generated text and been mostly limited to tasks such as machine translation [Malinin and

Gales, 2020], question answering [Kuhn et al., 2023], and text summarization [van der Poel et al., 2022]. However, the need for an uncertainty score for only a part of the generation substantially complicates the problem.

Another concern is that in the case of free-form LLM generations over open domains, there is no single gold standard reference answer that can be used to assess the factual correctness of a long free-text model output [Wang et al., 2024d]. This makes the factual evaluation of open-domain LLM responses non-trivial, either depending on manual verification or on automatic fact-checkers, e.g., FActScore and FacTool [Min et al., 2023, Chern et al., 2023]. Human assessment is costly [Nakov et al., 2021a, Chen et al., 2023], while existing automatic fact-checking systems are often unreliable. How to evaluate and to improve the accuracy of automated fact-checkers is critical for producing dependable LLM factuality evaluations.

Recent work on fact-checking compared the system predictions to human annotations. This only assesses the final verification results, i.e., whether a claim or a document is true or false [Chern et al., 2023, Chen et al., 2023, Dhuliawala et al., 2024]. Thus, the system is regarded as a black box. The lack of evaluation of the intermediate steps that the system has taken, makes it difficult to understand which specific system component(s) led to an erroneous factual judgement. Moreover, prior studies used disparate evaluations, rendering them hard to compare [Wang et al., 2024d].

To address these issues, this dissertation studies the development and use of efficient uncertainty estimation methods for solving applied Natural Language Processing tasks. Thus, we propose a batch active learning method based on the approximation of the BatchBALD method using pairwise mutual information to accelerate the process of acquisition of multiple training samples. To make our active learning method more accessible for industrial applications, we implemented it inside ALToolBox framework for efficient labeling and benchmarking of active learning strategies, and compared with other AL SOTA methods. Also, we propose an uncertainty estimation method for fact-checking generated output LLMs, which focuses on the uncertainty resulting from model uncertainty about the generated claim. Finally, we designed a fine-grained methodology for annotation for fact-checking, as well as constructed a holistic benchmark for the corresponding evaluation.

1.2 Research Questions

The aim of the presented thesis is to study and apply efficient uncertainty quantification methods for applied natural language processing tasks. Specifically, this thesis addresses the following research questions:

Q1: Can we design a method for quantifying uncertainty for batch active learning that can effectively choose multiple examples to label at each step?

Q2: Can we develop an uncertainty estimation method that distinguishes between different types of uncertainty in a generative language model?

Answering these research questions may allow for improved solutions to applied natural language processing tasks and potentially extend the knowledge gained to other research questions related to uncertainty estimation. These research questions will be discussed in detail in the following sections.

1.3 Contributions

This thesis has the following contributions:

1. We propose a new active learning algorithm called *Large BatchBALD (LBB)*, which is an approximation of the BatchBALD method that allows to improve computational efficiency, while keeping the diversity of samples in a batch (Section 3.1).

2. We benchmark our AL methods with a comprehensive collection of state-of-the-art query strategies via *ALToolBox* for text classification and image classification (Section 3.2).

3. We propose a new method for token-level uncertainty quantification *Claim Conditioned Probability (CCP)* that outperforms baselines and can be used as a plug-in in a fact-checking pipeline (Section 4.1).

4. We propose a *new framework* for fact-checking LLM generations using token-level uncertainty quantification (Sections 4.2, 4.3).

5. We provide an *empirical and ablation analysis* of the method for fact-checking of LLM generations to spot claims with factual errors for LLMs over multiple languages (Sections 4.4, 4.5).

6. We propose a holistic and fine-grained *framework for annotating the factuality of the outputs of LLMs*. (Section 5.1).

7. We construct *Factcheck-Bench*, a document-level claim-based fact-checking benchmark of LLMs, in terms of both detection and revision (Section 5.2).

1.4 Thesis Outline

This thesis is structured as follows.

Chapter 2 begins with an introduction to uncertainty quantification in applied NLP tasks and proceeds to an overview of uncertainty estimation methods in active learning. It continues with the background on fact-checking, canonical methods, and datasets used. The chapter concludes with an overview of uncertainty estimation methods in fact-checking.

Chapter 3 describes a new batch active learning algorithm Large BatchBALD. It gives a well-grounded approximation to the BatchBALD method. Thus, the new method aims to achieve comparable quality while being more computationally efficient. We provide a complexity analysis of the algorithm, showing a reduction in computation time, especially for large batches. Furthermore, we present an extensive set of experimental results on text and image data, utilizing ALToolBox benchmark.

Chapter 4 presents a new token-level uncertainty quantification method, Claim Conditioned Probability (CCP). It removes the impact of uncertainty about what claim to generate on the current step and what surface form to use, and measures only the uncertainty of a particular claim value expressed by the model. Experiments on the task of biography generation demonstrate strong improvements for CCP compared to the baselines for seven LLMs and four languages. Human evaluation reveals that the fact-checking pipeline based on uncertainty quantification is competitive with a fact-checking tool that leverages external knowledge.

Chapter 5 introduces a holistic end-to-end solution for annotating the factuality of LLM-generated responses, which encompasses a multi-stage annotation scheme designed to yield detailed labels concerning the verifiability and factual inconsistencies found in LLM outputs. We further construct an open-domain document-level factuality benchmark in three-level granularity: claim, sentence and document, aiming to facilitate the evaluation of automatic fact-checking systems.

Finally, in Chapter 6, we provide a discussion of the obtained results and outline future work directions.

Contribution note. This thesis includes the results of 2 papers with more than 10 contributors where the author is not the main/shared first author. Therefore, in this note, we additionally acknowledge the author's contributions to these papers. For the Factcheck-Bench work, the author was involved in developing a multi-stage annotation methodology and its implementation on real data (in other words, step-by-step manual annotation at all stages). As for the ALToolBox work, the author implemented and tested the LBB and PLBB methods proposed in his previous work on the benchmark of this work.

Chapter 2

Related Work

2.1 Uncertainty Quantification and Applications.

There are various applied tasks that rely to some extent on the intelligent development and application of uncertainty quantification methods: OOD detection [Yang et al., 2024], Bayesian optimization [Wang et al., 2023], active learning [Ren et al., 2021], conformal prediction [Angelopoulos and Bates, 2021], fact-checking [Guo et al., 2022], etc. There are different ways to measure uncertainty, each of which is effective in its own way. The question of the effectiveness of an uncertainty measure relates to how well the measure fits the conditions of the problem to be solved, how computationally efficient the method is, and what relevant properties it possesses.

In this thesis, we focus on the effectiveness of applying uncertainty estimation to two applied tasks: active learning and fact-checking. In the first task, uncertainty estimation is concerned with selecting the most informative data samples for model training, while in the second task, uncertainty estimation allows us to understand how reliable a model's output is with respect to some common knowledge. In each of these tasks, the effectiveness of the uncertainty method is inextricably linked to the problem formulation, as we will see later.

2.2 Uncertainty Quantification for Active Learning.

2.2.1 Introduction to Deep Active Learning

Uncertainty estimates in deep learning are often associated either with MC-dropout [Gal and Ghahramani, 2016] or deep ensembles [Lakshminarayanan et al., 2017]. In the case of MC-dropout, one samples the dropout mask on inference to get an ensemble of models differently parameterized. In the case of deep ensembles, one trains a single model with different weight initialization. Speaking of classification, in both scenarios, the final prediction is the average of the softmax vectors from all the models in the ensemble. In the work [Beluch et al., 2018] authors demonstrate the superior performance of ensembles over MC-dropout in image classification tasks.

One of the most applicable baseline algorithms for active learning is Bayesian Active Learning by Disagreement (BALD) [Houlsby et al., 2011]. Its acquisition function is computed as mutual information between model output and its latent parameters. That is, BALD tries to find those examples in which the different models disagree, while each of the models is confident in its prediction. This approach is quite efficient when taking one example at each step for training.

2.2.2 Batch Active Learning

In practice, with a large pool size, it is unprofitable to take a single example or even small batches, so it is important to be able to take batches of informative examples at each step of the AL loop. In practice, the top-k approach is most often used to take more than one example in the AL loop, where each step takes k examples with the highest values of the acquisition function. This approach has a serious drawback, namely, it does not take into account pairwise interaction of the data samples in batches that leads to acquiring a batch of similar examples and resulting performance degradation.

One possible solution is a batch modification of the BALD algorithm called BatchBALD [Kirsch et al., 2019]. The idea is to treat the mutual information in a batch manner as between a joint random variable (i. e., set of model outputs) and model parameters. In this scenario, points are added one at a time in a greedy manner, and the total mutual information is recursively recalculated. The diversity of acquired samples comes from accounting for the interactions in the batch between outputs. Nevertheless, BatchBALD has a significant drawback, namely, it takes a lot of working time [Kirsch et al., 2021]. In the original BatchBALD work, the standard choice is to take a batch of 10 elements, since complexity grows exponentially with batch size due to the joint entropy calculation. In practice, it is calculated directly for the first 5 samples in the batch, and the rest are sampled using MC-dropout.

Another way for diversification is presented in the already mentioned article [Kirsch et al., 2021]. Its authors get a diversity of chosen examples at the expense of the stochasticity of the acquisition function. Their idea is to get scores from some algorithm, such as BALD, and then translate those scores into a distribution. Sampling from such a distribution, we get an acquisition batch. Thus, if we take all the scores obtained with BALD, raise to some power, normalize, and then sample, we obtain the PowerBALD algorithm, which is one of the comparative baselines in this work. The basic idea is that a randomized sampling strategy is better than a greedy one, requires the same amount of time, and partially overcomes the data redundancy bottleneck.

2.2.3 Query Strategies

To achieve the best quality of AL algorithm, it is often useful to focus only on uncertainty that is directly related to the quality of the algorithm. Thus, the authors of MOCU-based algorithms [Zhao et al., 2021] propose to minimize only the classification error uncertainty as an acquisition function, taking into account the posterior, rather than the overall uncertainty, in contrast to BALD. As a result, the authors do not increase the probability of an already guessed classes, but extract controversial samples on the classification borders, taking into account the posterior. A similar idea was considered in the paper [Burnaev and Panov, 2015] where the authors proposed an acquisition function in a one-step look-ahead manner for regression on Gaussian processes [Rasmussen, 2006]. The idea is to choose a new point to add so that the average variance over the space is reduced. Strong uncertainty estimation methods based on softmax prediction probability also include methods such as Least Confidence (LC, [Lewis and Gale, 1994]), Maximum Normalized Log-Probability (MNLP) [Shen et al., 2017] and Breaking Ties (BT) [Luo et al., 2004].

Another important issue in active learning, affecting the performance of the model, is the diversity of acquired data samples. Combining a Bayesian network and a Gaussian process with a known covariance function, the authors in [Tsymbalov et al., 2019] propose to obtain diverse data samples from the acquisition function as the maximum variance of the Gaussian process. Also, since the variance of the Gaussian process does not depend on the output of the network, each time sampling a new point changes the total variance, which eliminates the need to retrain the network at each step. Another design of the acquisition function, which takes into account the diversity of samples, is based on the geometric properties of the data [Sener and Savarese, 2018]. The idea is to add images with the greatest distance from the training set to find data that is still poorly represented by the training set. Recently proposed Contrastive Active Learning (CAL) prioritizes instances, which predictive likelihoods diverge the most from their neighbors in the training set [Margatina et al., 2021].

Another natural but computationally expensive way to introduce diversity of AL samples into a training set is to use clustering. The authors in the paper [Citovsky et al., 2021] demonstrate an efficient data sampling with huge batch sizes by selecting samples from hierarchically clustered data in an ascending volume manner. In the work [Wan et al., 2021] the authors suggest using KNN classifier as the output layer of the network instead of softmax, due to better generalization ability to the unknown space. BERT-KM [Yuan et al., 2020] clusters texts in the unlabeled pool using their contextualized embeddings and selects nearest neighbors of cluster centers. Active Learning by Processing Surprisal (ALPS) [Yuan et al., 2020] leverages pre-trained models, self-supervised learning objective, and clustering to solve the cold-start problem in AL.

There are also several gradient-based query strategies. Expected Gradient Length (EGL) aims to prioritize instances that would impart the greatest change to the current model if we add them to the training set with their labels [Settles et al., 2007]. Batch Active Learning by Diverse Gradient Embeddings (BADGE) measures uncertainty as the gradient magnitude with respect to parameters in the final (output) layer [Ash et al., 2020]. Batch Active learning via Information Matrices (BAIT) selects batches of samples by optimizing a bound on the MLE error in terms of the Fisher information [Ash et al., 2021].

There are a number of difficulties in comparing active learning methods with each other. They are related to finding the best settings for all participating algorithms for a fair comparison. In addition, some algorithms are difficult to run out of the box and many benchmarks simply do not include them. Thus, Paladin [Nghiem et al., 2021], ActiveAnno [Wiechmann et al., 2021], and AlpacaTag [Lin et al., 2019] implement only the basic query strategies. FAMIE [Van Nguyen et al., 2022] implements several modern methods like ALPS and BADGE, but lacks many others.

Several low-level AL packages that focus on providing various query strategies and can be used as a building blocks for more elaborated systems: LibAct [Yang et al., 2017], ModAL [Danka and Horvath, 2018], Baal [Atighehchian et al., 2020], Small-text [Schröder et al., 2021]. However, most of them also overlook the problem of reusability and computational efficiency. Only Small-text is specifically tailored to NLP tasks.

2.3 Fact-checking Background

2.3.1 What is Fact-checking?

Fact-checking is the task of assessing whether claims made in writing are manipulated or true. This is typically broken down into the stages of claim detection, evidence retrieval, verdict prediction, and optionally justification prediction [Guo et al., 2022, Augenstein, 2021].

Claim detection is to identify claims that require verification, which commonly relies on the concept of check-worthiness. In the context of human-written documents, checkworthy claims are regarded as those for which the general public would be interested in knowing the truth [Hassan et al., 2015, Wright and Augenstein, 2020].

However, this may not be adaptable to machine-generated texts. Plausible hallucinations of LLMs make it difficult for general individuals to distinguish whether it is true or false, thus their outputs become less trustworthy than the statements made by humans. Current methods tend to check all factual claims of LLM generations [Chern et al., 2023].

Evidence retrieval aims to find sources supporting or refuting the claim. Claim verification is expected to assess the veracity of the claim and produce justification based on the retrieved evidence. That is, claims are assigned truthfulness labels, and explanations for verdicts are produced. A basic form of justification is to highlight the pieces of evidence used to reach a verdict [Guo et al., 2022].

Method	D	R	Granularity	Knowledge source	Datasets	Task	How_collect
Factcheck-GPT	✓	✓	claim	Google search	✓	Instruction	prompt ChatGPT and human annotation
FacTool [Chern et al., 2023]	✓	✓	article metadata	Google scholar	✓	Generate literature review	prompt ChatGPT
FacTool [Chern et al., 2023]	✓	✓	claim (gold)	Parsed Google search	RoSE/FactPrompts	Summarisation-eval/QA	human annotation: RoSE [Liu et al., 2023]
RARR [Gao et al., 2023]	✓	✓	document	Bing search	NQ,StrategyQA,QReCC	QA	human annotation
CoVe [Dhuliawala et al., 2024]	✓	✓	document	parametric knowledge	CoVe corpus	QA, instruction	human annotation
FELM [Chen et al., 2023]	✓	✗	segment	Google search	✓	Instruction	prompt ChatGPT and human annotate factuality
Self-contradictory [Mündler et al., 2024]	✓	✗	sentence	parametric knowledge	✓	Instruction	prompt ChatGPT,GPT-4 for contradictory sentence
SelfCheckGPT [Manakul et al., 2023]	✓	✗	sentence	parametric knowledge	✓	Generate Wikibio passage	prompt GPT3 and human annotate 3 factual labels
FACTOR [Muhlgay et al., 2023]	✓	✗	sentence	parametric knowledge	✓	Multichoice QA	prompt *davinci-003* for non-factual completions
HaluEval [Li et al., 2023]	✓	✗	document	parametric knowledge	✓	QA, summarise, dialogue	prompt ChatGPT to generate hallucinated answers
HaluEval [Li et al., 2023]	✓	✗	document	parametric knowledge	✓	Instruction	prompt ChatGPT, human annotate false segments
FActScore Min et al. [2023]	✓	✗	claim	Wiki Bio Generation	✓	Instruction	prompt ChatGPT to generate biography
FRESHQA [Vu et al., 2023]	✓	✗	facts in answer	parametric knowledge	✓	QA	collect questions with time-changing answers
Snowball [Zhang et al., 2024]	✓	✗	Yes/No answer	parametric knowledge	✓	QA	human annotation
SelfAware [Yin et al., 2023]	✓	✗	document	reference generations	✓	QA	collect unanswerable questions and prompt ChatGPT

Table 2.1: **Methods and benchmarks for hallucination Detection (D) and Revision (R)**. FacTool: article metadata is a tuple (paper title, year, authors). CoVe=Chain-of-Verification, CoVe corpus includes four existing datasets: Wikidata, Wiki-category, MultiSpanQA, and biographic. 3 labels in SelfCheckGPT: major/minor inaccurate and accurate. Unanswerable questions: the model should express uncertainty instead of delivering conclusive responses. FRESHQA collect four types of questions: false premise, answers never change, change slowly and fast over time.

2.3.2 Conventional Fact-checking

Previous works either focus on hallucinations in task-specific generations with references (to detect whether the generated output contradicts the source content), such as abstractive summarization [Maynez et al., 2020], machine translation [Raunak et al., 2021] and data-to-text generation [Liu et al., 2021], or concentrate on specific topics e.g. Covid-19 [Augenstein et al., 2019], politics [Barrera et al., 2020], climate [Bulian et al., 2020], and specific domains such as journalism, news, social media (e.g. Twitter [Pröllochs, 2022]) and Wikipedia (FEVER: Thorne et al. [2018]).

Moreover, most early studies only perform one or two subtasks in the factual error detection and correction, instead of the whole process. For example, many models estimate a label indicating whether the statement is supported or refuted by the evidence, given a (statement, evidence) pair as input [Thorne et al., 2018, Nie et al., 2019, Augenstein et al., 2019, Wadden et al., 2020]. To adapt to situations where relevant evidence for a statement is not readily available, some works explored how to automatically retrieve evidence that may help support or refute a statement [Fan et al., 2020, Nakov et al., 2021a, Gao et al., 2023].

More recent work has also explored how to correct claims based on retrieved evidence [Thorne and Vlachos, 2021, Schuster et al., 2021, Iv et al., 2022] and how to generate justification/explanation for verdicts on claims [Atanasova et al., 2020]. However, most factual correction used human-authored edits from FEVER [Thorne et al., 2018] as both their training and automatic evaluation data. FEVER's claims were extracted from Wikipedia. This limits the generalizability of these fact-checking models over generations of LLMs across various tasks and diverse domains.

2.3.3 LLM Fact-checking

The phenomenon that LLMs produce outputs that are seemingly plausible while deviating from the user input, previously generated context, or factual knowledge, is commonly referred to as hallucination [Zhang et al., 2023]. Based on the timing of the LLM life cycle, LLM hallucinations can be addressed during pretraining by automatically selecting reliable data or filtering out noisy data to mitigate hallucinations, in supervised fine-tuning by curating a small volume of high-quality training data, in reinforcement learning from human feedback (RLHF), and during inference by decoding strategies [Zhang et al., 2023]. We focus on approaches applied after inference.

Methods. For post-processing approaches to alleviating LLM hallucinations, recent studies can be roughly classified into two categories depending on whether they rectify errors: (1) detecting and correcting factual errors for free-form text; and (2) only detecting whether a text contain hallucinations (*Yes* or *No*). Both of them resort to either external knowledge or parametric knowledge to identify and rectify factually-incorrect statements [Gao et al., 2023, Chern et al., 2023, Manakul et al., 2023, Dhuliawala et al., 2024].

Though Self-contradictory [Mündler et al., 2024] involves revision, they aim to remove the conflicting information between the original sentence and the synthetically-generated contradictory sentence, instead of correcting factual errors in the original sentences. Given a LLM response, RARR [Gao et al., 2023] and CoVe [Dhuliawala et al., 2024] first generate a series of questions covering different aspects of the response, which serve as queries in the evidence retrieval, and then edit the whole response to correct factual errors. Such coarse granularity verification may miss out incorrect statements, particularly over long documents, and also makes it difficult to spot false spans precisely, thus disabling fine-grained (e.g., correct only a false number in a statement) and flexible edits (e.g., delete a completely-hallucinated sentence). Additionally, revising the whole document tends to result in poor preservation of the original input (e.g., style, vocabulary, and structure), introducing irrelevant descriptions and even new hallucinations.

Claim-level editing empowers precise correction and good preservation.
FacTool [Chern et al., 2023] performs fact-checking over claims. However, gold claims are required as input for the system. That is, users must first decompose an output from a LLM into a list of checkable atomic claims by themselves before using FacTool to check, which complicates the fact-checking process. Moreover, it is expensive to use FacTool to check a piece of text, since the whole checking process calls APIs including OpenAI ($0.06/1K tokens), Serper ($1.00/1k queries), and Scraper.[1] This also challenges the evaluation where online API is not allowed to call with the consideration of internal data protection.

Datasets. From the perspective of the evaluated benchmarks, as shown in Table 2.1, studies of the first category generally evaluate their methods on existing QA datasets,

[1] https://www.scraperapi.com/pricing/

Dataset	Granularity	Factual label	Revision	Length	Size
HaluEval	document	✓	✗	82.0	4,507
FELM-WK	segment	✓	✗	51.1	184
FactPrompts	claim	✓	✗	41.8	50
Factcheck-GPT	claim	✓	✓	73.1	94

Table 2.2: Statistics of world-knowledge factuality evaluation benchmarks. Length = the average number of words of LLM responses.

or revise hallucinations in a specific topic such as literature review and biographic generations [Chern et al., 2023, Dhuliawala et al., 2024]. These topics may not be frequently requested by general users in real-world scenarios.

Studies of the second category contribute a spectrum of benchmarks to detect diverse hallucinations, such as synthetically-generated contradictory sentences [Mündler et al., 2024], deliberately-generated hallucinated answers [Li et al., 2023] and non-factual completions given a prefix context [Muhlgay et al., 2023]. Manakul et al. [2023] manually annotate factual labels (major/minor inaccurate and accurate) given a sentence in the generated Wikibio passage.

Interestingly, Yin et al. [2023] collected 1,032 unanswerable questions from five diverse categories *no scientific consensus*, *imagination*, *completely subjective*, *too many variables*, *philosophical*, and their 2,337 answerable counterparts. Unanswerable questions refer to questions where the model should express uncertainty instead of delivering conclusive responses. Zhang et al. [2024] collected three datasets, with 500 questions (all *No* or all *Yes* answers) for each. One focuses on one type of question, including whether a number is a prime, senator search (whether a US city has a specific university), and whether there is a flight from one city to another given a graph connection.

However, these datasets are either only applicable in detection, or originate from a single task like biography writing [Min et al., 2023], without accounting for variations across different generations. HaluEval's annotation over Alpaca 5K responses of various instructions, which is one of the most similar works to ours. They ask human annotators to label whether the response contains hallucinated information (*Yes* or *No*) and list the corresponding spans if there exist errors [Li et al., 2023].[2] FacTool

[2] The hallucination is considered from the following three aspects: unverifiable, non-factual, and

evaluate over a knowledge-based QA dataset FactPrompts consisting of 50 (prompt, response) pairs. It is annotated by authors over atomic claims and their factual labels (true/false), but the responses tend to be short, instead of long documents (see Table 2.2). FELM [Chen et al., 2023] is only annotated with sentence-level *true or false* labels (no correction).

2.4 Fact-Checking via Uncertainty Quantification

2.4.1 Fact-Checking LLM Generations and Detecting Hallucinations

The problem of hallucinations has made fact-checking of LLM outputs a prominent topic in the research community and resulted in a surge of publications on the topic. Chern et al. [2023] present Factool – a task and domain agnostic framework for hallucination detection that leverages GPT for claim extraction and verification. Manakul et al. [2023] suggest to sample multiple outputs from black-box LLMs and evaluate how similar the sampled responses are using the external model. Varshney et al. [2023] detect LLM hallucinations by extracting key parts of output using an external model and estimating their uncertainty based on logits. The most uncertain parts are verified using an external knowledge source. Pan et al. [2023] propose to fact-check complex statements by decomposing them into simpler subtasks and generating reasoning programs to verify these statements. Min et al. [2023] present a methodology for evaluating long LLM-generated texts by decomposing them into simple atomic statements and further verifying them against some knowledge source. Several subsequent works further optimize components of the knowledge-based fact-checking pipelines [Wang et al., 2024a,c].

2.4.2 Uncertainty Quantification of LLM Generations

UQ techniques for LLM generation can be classified into five major categories [Fadeeva et al., 2023]. Information-based methods leverage the probability distribution of

irrelevant.

generated tokens and usually do not require any additional models. In this category, we can include methods such as perplexity [Fomicheva et al., 2020], mean token entropy [Fomicheva et al., 2020], point-wise mutual information (PMI) [Takayama and Arase, 2019], and conditional PMI [van der Poel et al., 2022].

Another category of methods is based on density estimation of latent instance representations. A typical example in this category is Mahalanobis distance [Lee et al., 2018] and its various modifications [Ren et al., 2023, Vazhentsev et al., 2023b]. The disadvantage of such methods is the need for access to the LLM training data in order to fit external density models, which is problematic for most general-purpose LLMs.

Ensembling and Monte Carlo dropout methods are based on the lexical diversity of multiple outputs sampled from one or multiple versions of LLMs for a single query [Malinin and Gales, 2020, Fomicheva et al., 2020]. Their main drawback is that they require many predictions, which makes them too computationally and memory intensive for practical purposes. Additionally, it is difficult to apply these techniques to quantify uncertainty of text fragments such as claims, as different samples may seriously diverge.

It has recently been shown that LLMs can reflexively estimate the confidence of their generations simply by asking themselves about the truthfulness of their output [Kadavath et al., 2022]. This can work better than analyzing the probability distribution for the original prediction, but requires a second pass of inference, feeding the original output as a part of the query.

Finally, there is a group of methods that leverages the diversity of meanings that the LLM generates for a given query. This group includes semantic entropy [Kuhn et al., 2023] and various scores based on the analysis of the similarity matrix between outputs [Lin et al., 2024].

UQ methods can also be classified into white-box and black-box [Lin et al., 2024] approaches, depending on the required access to LLM itself and its outputs. Black-box techniques do not require any other input except generated texts.

Chapter 3

Scalable Batch Acquisition for Deep Bayesian Active Learning

3.1 Methodology

3.1.1 Problem Setting

Here, we consider a pool-based active learning problem statement. This implies that we have a small amount of labeled examples $D_{\text{train}} = \{x_i, y_i\}_{i=1}^{N}$, $y_i \in \{1, \dots, C\}$ and a much larger pool of unlabeled data $D_{\text{pool}} = \{x_j\}_{j=1}^{N_{\text{pool}}}$. We consider a Bayesian model M with parameters $\theta \sim p(\theta \mid D_{\text{train}})$. Here, conditioning on D_{train} emphasizes that the model was trained using this dataset. Acquired samples from D_{pool} are selected as the ones that maximize a so-called acquisition function A:

$$x^* = \arg \max_{x \in D_{\text{pool}}} A(x \mid M, D_{\text{train}}). \tag{3.1}$$

It maps each example from the unlabeled pool to a numerical value using some information criteria or uncertainty measure. Acquired samples from the pool are selected for oracle labeling, and then these examples are added to the existing labeled data D_{train}. The target model is then trained on the currently labeled amount of data. This procedure is repeated throughout the active learning cycle and continues until the total budget of the algorithm is exhausted. At each step of the cycle, the acquisition function is recalculated. The quality of the model is evaluated on the test data D_{test} and compared at each step of the active learning procedure.

There are various acquisition functions applicable in AL. One of the standard ones is the Least Confident score:

$$A(x) = 1 - \max_{c \in C} p(y = c \mid x, \theta), \tag{3.2}$$

where θ are model parameters. That is, we choose a sample for which the model is most uncertain about the class predicted.

In this work, we use both MC-dropout and deep ensembles approaches for more accurate capturing of uncertainty for deep neural networks. In the case of deep ensembles, the models of the same architecture but trained from different initializations form an ensemble. In the case of MC-dropout, the dropout sampling on inference is used to obtain a set of networks with different parametrization using different dropout masks. Formally, in both cases the final prediction can be written as

$$\bar{p}_k(y = c \mid x, D_{\text{train}}) := \frac{1}{k} \sum_{i=1}^{k} p(y = c \mid x, \theta_i), \tag{3.3}$$

where θ_i is the model parameters for the i-th model and k is the number of models in a set. It can be either the number of network initializations in the case of deep ensemble, or the number of forward passes in the case of MC-dropout. Both MC-dropout and ensembling can be seen as instances of the general Bayesian formulation, with model parameters being samples from the posterior distribution: $\theta_i \sim p(\theta \mid D_{\text{train}}),\ i = 1, \ldots, k$.

While one can directly use the averaged predictive distribution (3.3) in the least confident acquisition function (3.2), it might be beneficial to extract some additional information from the posterior on top of the predictive mean. Next, we will focus on the family of entropy-based acquisition functions that allow to achieve that.

3.1.2 Entropy-based Acquisition Functions

Entropy

The entropy maximization criterion is also one of the basic ways of selecting examples for AL:

$$\mathrm{H}[y \mid x, D_{\text{train}}] = -\sum_{c=1}^{C} \bar{p}_k(y = c \mid x, D_{\text{train}}) \cdot \log \bar{p}_k(y = c \mid x, D_{\text{train}}). \tag{3.4}$$

It again uses the predictive mean only by looking at the entropy of this distribution. This criterion is maximized for samples having similar probabilities predicted for the different classes.

Bayesian Active Learning by Disagreement (BALD)

Starting from the entropy, one can construct criteria that look at the disagreement between the models in the Bayesian framework. The original BALD criterion [Houlsby et al., 2011] is formulated as the conditional mutual information $I(\theta, y \mid x, D_{\text{train}})$ between unknown (unobserved) output y and latent parameters θ, conditioned on input variable x and observed data D_{train}. Note, that the BALD criterion can be written in the y-space of mutual information and expressed as follows:

$$I_{\text{BALD}}(y; \theta) = \mathrm{H}[y \mid x, D_{\text{train}}] - \mathbb{E}_{\theta \sim p(\theta \mid D_{\text{train}})}\big[\mathrm{H}[y \mid x, \theta]\big],$$

where $\mathrm{H}[y \mid x, D_{\text{train}}]$ is an entropy of model output y conditioned on data sample x and train data, $\mathrm{H}[y \mid x, \theta]$ is an entropy of model output y conditioned on data sample x and sampled latent model parameters $\theta \sim p(\theta \mid D_{\text{train}})$ which are integrated out by the expectation $\mathbb{E}_{\theta \sim p(\theta \mid D_{\text{train}})}$. In other words, it calculates the difference between the entropy of marginal predictive distribution and posterior mean conditional entropy. BALD intuition is that it seeks for data samples in whose outputs y the model is the most uncertain (leads to high marginal entropy), while being certain about individual model parameters (leads to confident predictions but highly diverse). Being a special case of mutual information, BALD scores are always non-negative, and turn to zero at complete independence of model output y and latent parameters θ. Note that here and below we omit the conditionality of the mutual information I on x and D_train for convenience of notation.

In general, a continuous case BALD acquisition function is expressed as a KL divergence between $p(y; \theta)$ and $p(y)p(\theta)$:

$$I(y; \theta) = \int_\theta \int_y p(y; \theta) \log \frac{p(y; \theta)}{p(y)p(\theta)} \mathrm{d}y\mathrm{d}\theta. \tag{3.5}$$

In terms of batch active learning, when an acquisition batch where $y_{1:b} := y_1, \ldots, y_b$ consists of $b > 1$ data samples, the BALD score is the sum of individual scores for each of b objects in the batch:

$$I_{\mathrm{BALD}}(y_{1:b}; \theta) = \sum_{i=1}^{b} I(y_i; \theta). \tag{3.6}$$

This approach has a serious drawback, namely, it does not take into account pairwise interactions of the data samples in batches. As a result, BALD tends to acquire a batch of similar examples, leading to suboptimal performance.

BatchBALD

To diversify samples in a batch, the BatchBALD acquisition function was proposed [Kirsch et al., 2019]. It is formulated as mutual information between a batch of observations and latent parameters:

$$I_{\mathrm{BB}}(y_{1:b}; \theta) = \mathrm{H}[y_1, \ldots, y_b \mid x_1, \ldots, x_b, D_{\mathrm{train}}] - \mathbb{E}_{\theta \sim p(\theta|D)} \big[\mathrm{H}[y_1, \ldots, y_b \mid x_1, \ldots, x_b, \theta]\big], \tag{3.7}$$

where $y_{1:b} := y_1, \ldots, y_b$ is treated as a joint random variable, b is the batch acquisition size. BatchBALD calculates mutual information between model output and model parameters but in a batch sense, that is, considering inter-variable correlation and taking a batch of outputs as a joint random variable. It allows to account for variable interconnections and provides diverse data sampling. In the continuous case, it is

BALD	BatchBALD	Large BatchBALD (ours)
$\mathcal{O}((b+k)\cdot\lvert D_{\text{pool}}\rvert)$	$\mathcal{O}(b\cdot c\cdot\min\{c^b,m\}\cdot\lvert D_{\text{pool}}\rvert\cdot k)$	$\mathcal{O}(\lvert D_{\text{pool}}\rvert^2\cdot k\cdot c+\lvert D_{\text{pool}}\rvert\cdot(b+k))$

Table 3.1: Complexity of BALD-based algorithms, where c is the number of classes, b is the batch acquisition size, k is the number of MC-dropout samples, m is the number of MC-sampled output configurations $y_{1:b-1}$.

again given by the corresponding KL divergence:

$$I(y_{1:b};\theta) = \int_\theta \int_{y_{1:b}} p(y_{1:b};\theta)\log\frac{p(y_{1:b};\theta)}{p(y_{1:b})p(\theta)}\mathrm{d}y_{1:b}\mathrm{d}\theta. \tag{3.8}$$

While accounting nicely for the correlation between observations, BatchBALD criterion is often computationally expensive, especially for large batches (see Section 3.1.4 for the complexity analysis). In the next section, we are going to provide its more computationally feasible alternative.

3.1.3 Large BatchBALD

One of the possible generalizations of mutual information is total correlation [Watanabe, 1960] between b random variables, which is defined as:

$$C(y_{1:b}) = \int\limits_{y_{1:b}} p(y_{1:b})\log\frac{p(y_{1:b})}{\prod_{i=1}^b p(y_i)}\mathrm{d}y_{1:b}. \tag{3.9}$$

It measures inter-variable dependencies, always positive and is nullified if and only if all the variables are independent of each other. Note that its form doesn't include model latent parameters θ.

The main idea of introducing of total correlation in this work is that it is exactly equal to the difference between BALD and BatchBALD acquisition functions:

$$\sum_{i=1}^b I(y_i;\theta) - I_{\text{BB}}(y_{1:b};\theta) - C(y_{1:b}) = 0, \tag{3.10}$$

where $C(y_{1:b}) = C(y_1; y_2; \ldots; y_b)$ is the mutual information of b random variables (i. e., generalization of mutual information of two variables), b is an acquisition batch size.

Calculation of total correlation and, therefore, calculation of BatchBALD can be extremely time-consuming. However, one can overcome this issue inspiring from another possible form of total correlation [Srinivasa, 2005] that uses mutual information of all possible variable subscripts:

$$C(y_{1:b}) = \sum_{i \neq j}^{b} I\left(y_i; y_j\right) - \sum_{i \neq j \neq k}^{b} I\left(y_i; y_j; y_k\right) + \ldots + (-1)^b I\left(y_1; y_2; \ldots; y_b\right). \tag{3.11}$$

The idea is to neglect higher order terms and use total correlation approximation with only pairwise mutual information components:

$$\hat{C}(y_{1:b}) = \sum_{i=1}^{b} \sum_{j \neq i}^{b} I\left(y_i; y_j\right). \tag{3.12}$$

Using the approximation of total correlation, we obtain

$$I_{\mathrm{BB}}(y_{1:b}; \theta) \approx \sum_{i=1}^{b} I(y_i, \theta) - \hat{C}(y_{1:b}). \tag{3.13}$$

We denote this approximation as Large BatchBALD (LBB), and the resulting formula for it becomes

$$I_{\mathrm{LBB}}(y_{1:b}; \theta) := \sum_{i=1}^{b} I(y_i, \theta) - \sum_{i=1}^{b} \sum_{j \neq i}^{b} I(y_i; y_j). \tag{3.14}$$

Importantly, LBB is significantly less computationally expensive compared to BatchBALD, see the complexity analysis in Section 3.1.4.

Main Property Proof. Let us prove the following statement:

$$\sum_{i=1}^{b} I(y_i, \theta) - I(y_{1:b}; \theta) = C(y_{1:b}). \tag{3.15}$$

Mutual information (aka BALD acquisition function) can be written by definition:

$$I(y_i; \theta) = \int_{\theta} \int_{y_i} p(y_i; \theta) \log \frac{p(y_i; \theta)}{p(y_i) p(\theta)} \mathrm{d}y_i \mathrm{d}\theta = \int_{\theta} \int_{y_i} p(y_i; \theta) \log \frac{p(y_i \mid \theta)}{p(y_i)} \mathrm{d}y_i \mathrm{d}\theta. \tag{3.16}$$

Mutual information of a joint random variable (aka BatchBALD acquisition function) can be expressed as:

$$
\begin{aligned}
I(y_{1:b};\theta) &= \int_\theta \int_{y_{1:b}} p(y_{1:b};\theta) \log \frac{p(y_{1:b};\theta)}{p(y_{1:b})p(\theta)} \,\mathrm{d}y_{1:b}\,\mathrm{d}\theta \\
&= \int_\theta \int_{y_{1:b}} p(y_{1:b} \mid \theta)p(\theta) \log \frac{p(y_{1:b} \mid \theta)}{p(y_{1:b})} \,\mathrm{d}y_{1:b}\,\mathrm{d}\theta \\
&= \int_\theta \int_{y_{1:b}} p(y_{1:b} \mid \theta)p(\theta) \log p(y_{1:b} \mid \theta) \,\mathrm{d}y_{1:b}\,\mathrm{d}\theta \\
&- \int_\theta \int_{y_{1:b}} p(y_{1:b} \mid \theta)p(\theta) \log p(y_{1:b}) \,\mathrm{d}y_{1:b}\mathrm{d}\theta \\
&= \int_\theta \int_{y_{1:b}} p(y_{1:b} \mid \theta)p(\theta) \log p(y_{1:b} \mid \theta) \,\mathrm{d}y_{1:b}\,\mathrm{d}\theta \\
&- \int_{y_{1:b}} p(y_{1:b}) \log p(y_{1:b}) \,\mathrm{d}y_{1:b}.
\end{aligned}
$$

Mutual information of b random variables one can write as:

$$
\begin{aligned}
&C(y_{1:b}) = \int\limits_{y_1:b} p(y_{1:b}) \log \frac{p(y_{1:b})}{\prod\limits_{i=1}^{b} p(y_i)} \,\mathrm{d}y_{1:b} \\
&= \int\limits_{y_{1:b}} p(y_{1:b}) \log p(y_{1:b}) \,\mathrm{d}y_{1:b} - \int\limits_{y_{1:b}} p(y_{1:b}) \log \prod_{i=1}^{b} p(y_i) \,\mathrm{d}y_{1:b} \\
&= \int\limits_{y_{1:b}} p(y_{1:b}) \log p(y_{1:b}) \,\mathrm{d}y_{1:b} - \sum_{i=1}^{b} \int\limits_{y_{1:b}} p(y_{1:b} \mid \theta)p(\theta) \log p(y_i) \,\mathrm{d}y_{1:b} \\
&= \int\limits_{y_{1:b}} p(y_{1:b}) \log p(y_{1:b}) \,\mathrm{d}y_{1:b} - \sum_{i=1}^{b} \int\limits_{y_i} p(y_i) \log p(y_i) \,\mathrm{d}y_i.
\end{aligned}
$$

Using the equations above, we can write the following:

$$
\sum_{i=1}^{b} I(y_i;\theta) - I(y_{1:b};\theta) - C(y_{1:b}) \tag{3.17}
$$

(by definition)

$$
\begin{aligned}
&= \sum_{i=1}^{b} \int_\theta \int_{y_i} p(y_i;\theta) \log \frac{p(y_i;\theta)}{p(y_i)p(\theta)} \,\mathrm{d}y_i - \int_\theta \int_{y_{1:b}} p(y_{1:b};\theta) \log \frac{p(y_{1:b};\theta)}{p(y_{1:b})p(\theta)} \,\mathrm{d}y_{1:b} \\
&- \int_{y_{1:b}} p(y_{1:b}) \log \frac{p(y_{1:b})}{\prod\limits_{i=1}^{b} p(y_i)} \,\mathrm{d}y_{1:b}
\end{aligned}
$$

(reduce the numerator and denominator)

$$=\sum_{i=1}^{b}\int_{\theta}\int_{y_i} p(y_i;\theta)\log\frac{p(y_i\mid\theta)}{p(y_i)}\,\mathrm{d}y_i-\int_{\theta}\int_{y_{1:b}} p(y_{1:b};\theta)\log\frac{p(y_{1:b}\mid\theta)}{p(y_{1:b})}\,\mathrm{d}y_{1:b}$$
$$-\int_{y_{1:b}} p(y_{1:b})\log\frac{p(y_{1:b})}{\prod_{i=1}^{b}p(y_i)}\,\mathrm{d}y_{1:b}$$

(factorization of joint probability due toindependence of y_i conditioned on θ)

$$=\sum_{i=1}^{b}\int_{\theta}\int_{y_i} p(y_i;\theta)\log\frac{p(y_i\mid\theta)}{p(y_i)}\,\mathrm{d}y_i-\int_{\theta}\int_{y_{1:b}} p(y_{1:b};\theta)\log\frac{\prod_{i=1}^{b}p(y_i\mid\theta)}{p(y_{1:b})}\,\mathrm{d}y_{1:b}$$
$$-\int_{y_{1:b}} p(y_{1:b})\log\frac{p(y_{1:b})}{\prod_{i=1}^{b}p(y_i)}\,\mathrm{d}y_{1:b}$$

(log of product is equal to sum of logs

and log fraction is expressed as a diff. of logs)

$$=\sum_{i=1}^{b}\int_{\theta}\int_{y_i}\big(p(y_i\mid\theta)p(\theta)\log p(y_i\mid\theta)-p(y_i\mid\theta)p(\theta)\log p(y_i)\big)\,\mathrm{d}y_i\,\mathrm{d}\theta$$
$$-\sum_{i=1}^{b}\int_{\theta}\int_{y_{1:b}}\big(p(y_{1:b}\mid\theta)p(\theta)\log p(y_i\mid\theta)-p(y_{1:b}\mid\theta)p(\theta)\log p(y_{1:b})\big)\,\mathrm{d}y_{1:b}\,\mathrm{d}\theta$$
$$-\int_{y_{1:b}} p(y_{1:b})\log\frac{p(y_{1:b})}{\prod_{i=1}^{b}p(y_i)}\,\mathrm{d}y_{1:b}$$

$$\left(\text{integr. out: } p(y_{1:b};\theta)=\prod_{i=1}^{b}p(y_i\mid\theta)p(\theta)\right)$$

$$=\sum_{i=1}^{b}\int_{\theta}\int_{y_i} p(y_i\mid\theta)p(\theta)\log p(y_i\mid\theta)\,\mathrm{d}y_i\,\mathrm{d}\theta-\sum_{i=1}^{b}\int_{y_i}p(y_i)\log p(y_i)\,\mathrm{d}y_i$$
$$-\sum_{i=1}^{b}\int_{\theta}\int_{y_i} p(y_i\mid\theta)p(\theta)\log p(y_i\mid\theta)\,\mathrm{d}y_i\,\mathrm{d}\theta+\int_{y_{1:b}}p(y_{1:b})\log p(y_{1:b})\,\mathrm{d}y_{1:b}$$
$$-\int_{y_{1:b}}p(y_{1:b})\log p(y_{1:b})\,\mathrm{d}y_{1:b}+\sum_{i=1}^{b}\int_{y_i}p(y_i)\log p(y_i)\,\mathrm{d}y_i=0.$$

which is an exact statement that was presented in the beginning.

Pairwise Mutual Information Calculation. Here we give an explicit formula for calculating the pairwise mutual information $I(y_i; y_l)$:

$$I(y_i; y_l) = \sum_{\hat{y}_i, \hat{y}_l = 1}^{C} \left[\frac{1}{k} \sum_{j=1}^{k} p\left(\hat{y}_i \mid \hat{\theta}_j\right) p\left(\hat{y}_l \mid \hat{\theta}_j\right) \right] \cdot \left[\log \left(\frac{1}{k} \sum_{j=1}^{k} p\left(\hat{y}_i \mid \hat{\theta}_j\right) p\left(\hat{y}_l \mid \hat{\theta}_j\right) \right) - \log \left(\frac{1}{k} \sum_{j=1}^{k} p\left(\hat{y}_i \mid \hat{\theta}_j\right) \right) - \log \left(\frac{1}{k} \sum_{j=1}^{k} p\left(\hat{y}_l \mid \hat{\theta}_j\right) \right) \right],$$

where $i \neq l$, and k is either the number of forward passes in the case of MC-dropout, or the number of models in an ensemble in the case of deep ensembles.

That is, it consists of two tensor multiplication operations with dimensionality $[n, k, C] \times [n, k, C] = [n, n, k, C]$ and $[n, C] \times [n, C] = [n, n, C]$, where n is a processing batch size.

3.1.4 Computational Complexity

In this section, we discuss the computational complexity for the BALD, BatchBALD, and Large BatchBALD algorithms, see Table 3.1. We also give some intuition how LBB, using BALD and pairwise mutual information, can significantly improve the complexity of the BatchBALD algorithm, especially noticeable in the case of large batches.

BALD

BALD time complexity is $\mathcal{O}((b+k) \cdot |D_{\text{pool}}|)$ and consists of calculating for each element of D_{pool} 2 components: $\mathcal{O}(b \cdot |D_{\text{pool}}|)$ is a cost to compute predictive distribution and $\mathcal{O}(k \cdot |D_{\text{pool}}|)$ is a cost to compute entropies in output space. It can be described as follows. For every element of a batch of size b, from a data pool $|D_{\text{pool}}|$ with c possible classes, a new data sample is searched as a maximum of difference between joint entropy and conditional joint entropy.

BatchBALD

BatchBALD complexity is $\mathcal{O}(b \cdot c \cdot \min\{c^b, m\} \cdot |D_{\text{pool}}| \cdot k)$, where c is the number of classes, k is the number of MC-dropout samples, and m is the number of MC-sampled configurations of $y_{1:b-1}$, $|D_{\text{pool}}|$ is a volume of unlabeled pool data. It can be described as follows. On each of $i = 1 : b$ the acquisition steps, a new candidate x_i with $p(y_i \mid \theta)$ from $|D_{\text{pool}}|$ is greedily selected to the already formed batch $p(y_{1:i-1} \mid \theta)$ of elements $x_{1:i-1}$. This batch is already calculated and stored, elements $x_{1:i-1}$ are fixed, so the task is to calculate joint entropy between a new added point x_i and an existed batch in a one by one manner. In exact (means based on given draws of θ) joint entropy scenario, all possible combinations $y_{1:i-1}$ can be calculated exactly as c^i meaning c possible classes of each of i elements in a batch. As for the approximated joint entropy scenario, if c^i value is big (in BatchBALD paper it is assumed after 5 acquired elements) then $p(y_{1:i-1})$ of $y_{1:i-1}$ is approximated using m MC-samples. In both cases, joint probability $p(y_1, \ldots, y_i)$ is calculated by averaging over $p(y_1, \ldots, y_i \mid \theta)$ (i. e., to find a probability density marginalizing over θ) with k MC-dropouts of θ. So, batch is selected in linear time, although joint probability still requires a lot of computational resources both in exact and approximate setting. For more details, see Kirsch et al. [2019].

Large BatchBALD

Large BatchBALD time complexity is equal to the sum of BALD complexity and total correlation approximation complexity. BALD complexity, as noted above, is $\mathcal{O}((b+k) \cdot |D_{\text{pool}}|)$. Total correlation approximation complexity is $\mathcal{O}\left(2 \cdot \frac{|D_{\text{pool}}| \cdot |D_{\text{pool}} - 1|}{2} \cdot k \cdot c\right) = \mathcal{O}\left(|D_{\text{pool}}|^2 \cdot k \cdot c\right)$. It means, that Large BatchBALD complexity is $\mathcal{O}(|D_{\text{pool}}|^2 \cdot k \cdot c + |D_{\text{pool}}| \cdot (b+k))$.

The resulting complexity has the following intuition behind. The given asymptotics denotes a linear dependence on the size of the batch, as in BALD. Thus, Large BatchBALD scales to the size of a batch consisting of hundreds of elements without significant costs. At the same time, the original BatchBALD algorithm works in a reasonable time only with batches consisting of tens of elements due to the calculation of mutual information of a joint random variable. In general, adding large batches in an active learning problem is a common practical scenario. With a huge pool of unlabeled data, adding a small amount of data up to a few tens will have little effect on the performance of the final model. Thus, it is computationally more efficient to be able to acquire large batches for annotation and training. Furthermore, LBB works equally well with batches of tens of elements and already shows computational superiority in comparison with BatchBALD, see Table 3.6.

3.1.5 Beyond Greedy Approaches via Sampling

While BatchBALD and its modifications are efficient in obtaining diverse batches, one can propose alternative strategies to achieve a similar effect. One natural way is to step aside from greedy sampling and introduce stochasticity into the procedure. The idea is to convert the resulting scores to a distribution and then sample from it. In this case, we raise the scores to some power $\alpha > 0$ and normalize the resulting values. Thus, the probability of selecting a sample x with an acquisition function equal to $A(x)$ from the unlabeled pool D_{pool} is

$$p_\alpha(x) = \frac{A^\alpha(x)}{\sum\limits_{x_i \in D_{\text{pool}}} A^\alpha(x_i)}. \tag{3.18}$$

In this work we consider such extension for the LBB and BALD algorithms, and call them Power Large BatchBALD (PLBB) and PowerBALD (PBALD) [Kirsch et al., 2021]. Thus, in the case of PLBB, the acquisition function is $A(x) = I_{\text{LBB}}(y; \theta)$, and in the case of PBALD it is $A(x) = I_{\text{BALD}}(y; \theta)$. Here y is the model output on the sample x and θ is the vector of model parameters. The magnitude of the power α in this case determines how much of a stochastic effect is present: with a smaller power the random effect is greater, with a greater power examples with larger scores are even more likely to be taken, and the random effect appears less.

3.2 Experiments

3.2.1 Text-domain Experiments

First, experimentally studied the proposed method on textual data. We used the ALToolBox benchmark [Tsvigun et al., 2022] for scalability and extended comparison with state-of-the-art active learning methods. For this purpose, we embedded our LBB and PLBB algorithms as well as our main competitor PBALD inside the ALToolBox framework[1] and performed an experimental comparison on several datasets. We considered AG News [Zhang et al., 2015], CoLA [Warstadt et al., 2019] and IMDb [Maas et al., 2011]. AG News is a large dataset including 120K sentences from news articles attributed to 4 classes. IMDb is a movie review dataset with 50K two-class examples. CoLA (Corpus of Linguistic Acceptability) is a dataset of linguistic publications, including more than 10K sentences from 2 classes. In this series of experiments, we use the DistilBERT [Sanh et al., 2019] model as a one capable of obtaining acceptable quality with minimal runtime. Here, we use MC-dropout to estimate uncertainty. All results are presented as the average of 5 runs with different random seeds with the corresponding standard deviation (for figures it is expressed as error bars, for tables it is mean ± std).

AG-News

The results on AG News dataset show that the proposed PLBB algorithm shows the best quality among the competitors, see Figure 3-1. Moreover, even the original LBB performs better than the naive baselines, as well as BALD and BatchBALD. Note that this behavior is noticeable even on a small 10-sample batch size. There are also results on a bigger batch size and with more competitors, see Section A.1.

Also, we made a comprehensive comparison with SOTA methods from ALToolBox benchmark for bigger acquisition batch size 50 and 40 starting data samples, see Table 3.2. We see that PLBB is one of three main leaders, along with BADGE and Least Confidence.

IMDb

Here, we attach experimental results on another text dataset IMDb for both small (10) and big (100) acquisition batches, see Tables 3.3 and 3.4. Corresponding number of starting points is 10 and 100, respectively. The number of classes is 2, initial data is balanced. We see that for the small batch, the stochastic extension of our algorithm Power Large BatchBALD outperforms alternatives in terms of average

[1] https://github.com/AIRI-Institute/al_toolbox

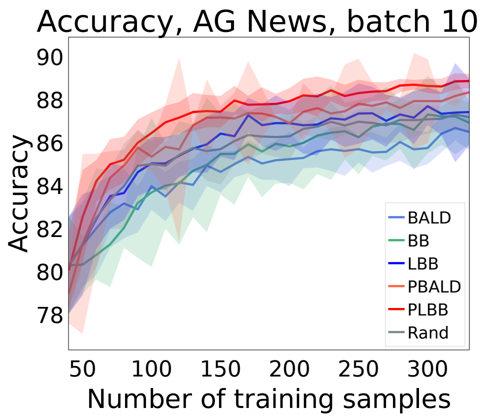


Figure 3-1: Performance comparison of AL algorithms on AG News dataset. PLBB (ours) clearly outperform other competitors.

AL Strategy	**Iter. 1**	**Iter. 5**	**Iter. 10**	**Iter. 15**	**Average**
ALPS	83.91 ± 1.7	87.07 ± 0.9	88.12 ± 0.6	88.32 ± 0.7	86.89 ± 0.7
BADGE	**85.6 ± 1.6**	**89.19 ± 0.3**	**89.99 ± 0.2**	**90.42 ± 0.3**	**89.85 ± 0.4**
BALD	82.71 ± 1.0	86.65 ± 0.3	88.12 ± 0.4	88.93 ± 0.3	88.28 ± 0.4
BatchBALD (BB)	84.14 ± 1.2	87.58 ± 0.9	88.74 ± 0.5	89.36 ± 0.1	88.54 ± 0.4
Breaking Ties (BT)	**85.98 ± 0.5**	**89.16 ± 0.1**	**89.95 ± 0.2**	**90.55 ± 0.2**	88.75 ± 0.3
CAL	**85.13 ± 1.1**	**88.62 ± 0.2**	**89.31 ± 0.4**	89.91 ± 0.4	88.14 ± 0.5
Coreset	83.28 ± 1.7	**88.48 ± 0.4**	**89.47 ± 0.3**	89.69 ± 0.2	87.96 ± 0.6
Entropy	84.47 ± 1.2	87.38 ± 1.4	**89.12 ± 0.4**	89.38 ± 0.2	87.42 ± 0.9
Least Confidence (LC)	**85.34 ± 0.9**	**88.87 ± 0.5**	**89.53 ± 0.5**	**90.17 ± 0.3**	**89.45 ± 0.4**
Large BatchBALD (LBB, ours)	84.7 ± 1.2	87.9 ± 1.7	88.52 ± 0.9	89.41 ± 0.6	88.47 ± 1.0
PowerBALD (PBALD)	83.04 ± 1.5	87.84 ± 0.5	**89.11 ± 0.2**	89.52 ± 0.4	88.83 ± 0.4
Power LBB (PLBB, ours)	**85.72 ± 1.5**	**88.4 ± 0.5**	**89.26 ± 0.2**	**90.19 ± 0.2**	**89.39 ± 0.4**
Random	84.12 ± 1.7	87.1 ± 0.4	87.98 ± 0.3	88.49 ± 0.1	87.86 ± 0.4

Table 3.2: Accuracy of DistilBERT on AG News with various AL strategies on several AL iterations with query size = 50 instances. *Average* refers to the average result throughout the AL cycle. We select with **bold** state-of-the-art results with respect to confidence intervals. The results are averaged for 5 runs with different seeds to ensure the stability.

performance and iteration-wise quality.

AL Strategy	**Iter. 1**	**Iter. 5**	**Iter. 10**	**Iter. 15**	**Iter. 20**	**Iter. 25**	**Iter. 30**	**Average**
BADGE	61.5 ± 4.4	79.04 ± 2.6	83.03 ± 1.0	82.75 ± 1.9	84.54 ± 0.6	84.76 ± 0.8	83.33 ± 3.0	**81.06 ± 1.9**
BALD	59.67 ± 4.4	72.34 ± 9.6	83.21 ± 0.9	83.34 ± 1.5	**85.8 ± 0.2**	**86.4 ± 0.5**	**86.14 ± 1.1**	80.12 ± 2.5
CAL	56.43 ± 0.4	67.47 ± 4.1	**87.09 ± 7.5**	83.68 ± 0.5	84.13 ± 0.8	85.46 ± 1.1	85.5 ± 0.7	78.79 ± 2.1
Coreset	60.18 ± 5.0	79.67 ± 2.0	81.42 ± 1.0	82.73 ± 1.6	84.05 ± 0.9	84.31 ± 1.0	84.64 ± 0.7	80.4 ± 1.7
LBB (ours)	62.71 ± 6.1	72.31 ± 5.7	79.11 ± 5.0	**84.35 ± 1.3**	**85.01 ± 0.7**	85.18 ± 0.7	85.99 ± 0.2	79.71 ± 3.2
LC	65.83 ± 6.7	74.2 ± 3.5	83.31 ± 0.9	82.6 ± 3.2	84.78 ± 0.6	84.92 ± 1.3	85.96 ± 0.2	80.46 ± 2.3
PBALD	58.3 ± 3.2	67.7 ± 0.3	83.71 ± 0.1	81.2 ± 0.3	84.34 ± 0.5	85.37 ± 0.3	85.64 ± 0.8	78.91 ± 1.5
PLBB (ours)	**66.09 ± 4.9**	**78.68 ± 2.3**	84.17 ± 1.0	**84.8 ± 0.3**	**85.07 ± 0.4**	83.02 ± 3.5	85.3 ± 1.2	**81.47 ± 2.1**
Random	64.09 ± 8.7	72.29 ± 5.9	83.52 ± 0.8	83.66 ± 0.2	84.7 ± 0.6	85.28 ± 0.2	85.5 ± 0.4	80.24 ± 2.0

Table 3.3: Accuracy of DistilBERT on IMDb with various AL strategies on several AL iterations with query size = 10. *Average* refers to the average result throughout the AL cycle. We select with **bold** state-of-the-art results with respect to confidence intervals. The results are averaged for 5 runs with different seeds to ensure the stability.

For the more complex setup with the batch of 100 of data samples, we see that

LBB and PLBB are iteration-wise leaders, while the original BALD is the leader in terms of average performance.

AL Strategy	Iter. 1	Iter. 5	Iter. 10	Iter. 15	Iter. 20	Average
BADGE	84.32 ± 0.42	86.92 ± 0.71	87.97 ± 0.29	88.81 ± 1.13	89.34 ± 0.42	87.82 ± 0.36
BALD	84.82 ± 0.6	**87.2 ± 0.32**	**88.67 ± 0.2**	88.79 ± 0.51	89.35 ± 0.14	**88.09 ± 0.13**
CAL	84.13 ± 1.28	86.72 ± 0.55	**88.29 ± 0.38**	88.93 ± 0.32	88.86 ± 0.76	87.67 ± 0.46
Coreset	84.1 ± 0.52	86.38 ± 0.89	87.63 ± 0.5	88.47 ± 0.42	89.1 ± 0.31	87.46 ± 0.33
LBB (ours)	83.44 ± 2.8	**87.29 ± 0.2**	**88.41 ± 0.4**	88.85 ± 0.2	**89.43 ± 0.1**	87.59 ± 0.6
LC	83.92 ± 1.38	86.47 ± 1.16	**88.41 ± 0.54**	**89.19 ± 0.32**	89.29 ± 0.18	87.74 ± 0.35
PBALD	**85.12 ± 0.7**	86.58 ± 1.8	87.82 ± 1.1	88.96 ± 0.2	89.36 ± 0.3	87.64 ± 0.7
PLBB (ours)	81.68 ± 6.5	**87.19 ± 0.2**	**88.16 ± 0.4**	88.97 ± 0.2	**89.56 ± 0.1**	87.61 ± 0.6
Random	82.81 ± 3.7	85.77 ± 0.61	86.84 ± 0.51	87.45 ± 0.33	88.06 ± 0.49	86.56 ± 0.5

Table 3.4: Accuracy of DisitlBERT on IMDB with various AL strategies on several AL iterations with query size = 100. *Average* refers to the average result throughout the AL cycle. We select with **bold** state-of-the-art results with respect to confidence intervals. The results are averaged for 5 runs with different seeds to ensure the stability.

CoLA

Also, we attach CoLA experimental results for the batch 10, see Table 3.5. We see that Coreset is the leader in terms of both iteration-wise and average performance, while PLBB and BADGE are close competitors to it in terms of iteration-wise performance.

AL Strategy	Iter. 1	Iter. 5	Iter. 10	Iter. 15	Iter. 20	Iter. 25	Iter. 30	Average
BADGE	63.66 ± 4.45	67.17 ± 2.53	69.64 ± 1.61	71.74 ± 1.63	**72.69 ± 1.06**	**73.08 ± 0.64**	**73.68 ± 0.53**	**70.63 ± 1.28**
BALD	59.39 ± 7.76	64.93 ± 4.71	68.4 ± 3.55	70.32 ± 1.18	71.2 ± 1.19	72.24 ± 1.65	73.2 ± 3.92	69.23 ± 9.05
CAL	60.2 ± 7.09	**69.11 ± 1.3**	**71.02 ± 1.27**	71.82 ± 1.32	72.3 ± 1.37	72.86 ± 1.06	**73.64 ± 0.79**	70.81 ± 0.88
Coreset	63.23 ± 6.22	**69.91 ± 1.63**	**71.29 ± 1.16**	**72.13 ± 1.03**	**72.99 ± 0.67**	**73.43 ± 0.54**	**73.92 ± 0.61**	**71.68 ± 0.67**
LBB (ours)	**67.04 ± 1.57**	65.18 ± 3.6	69.57 ± 1.14	71.51 ± 1.0	71.97 ± 1.06	72.77 ± 0.9	72.79 ± 0.7	70.22 ± 1.7
LC	61.19 ± 6.3	67.33 ± 2.37	70.04 ± 1.59	71.4 ± 0.92	71.77 ± 0.86	72.49 ± 0.99	73.01 ± 0.85	70.23 ± 1.22
PBALD	64.31 ± 5.07	68.32 ± 2.6	68.51 ± 2.62	69.93 ± 2.36	71.79 ± 1.23	72.89 ± 0.9	73.1 ± 0.9	69.76 ± 2.2
PLBB (ours)	**68.7 ± 1.34**	68.78 ± 0.91	70.89 ± 1.6	**72.61 ± 1.4**	**72.83 ± 1.1**	**73.35 ± 0.7**	**73.44 ± 0.7**	**70.95 ± 1.0**
Random	56.86 ± 7.72	64.13 ± 2.11	67.72 ± 1.98	69.96 ± 1.77	71.19 ± 0.96	72.08 ± 0.63	71.8 ± 1.34	68.53 ± 1.02

Table 3.5: Accuracy of DistilBERT on CoLA with various AL strategies on several AL iterations with query size = 10. *Average* refers to the average result throughout the AL cycle. We select with **bold** state-of-the-art results with respect to confidence intervals. The results are averaged for 5 runs with different seeds to ensure the stability.

3.2.2 Image-domain Experiments

For all datasets, the initial training set is balanced by the number of samples of each class. All datasets with the repetition option use each incoming object more than once (4 in our case) with a small Gaussian noise applied.

After each addition of new samples to the training set, the network is trained from scratch. All models use Glorot initialization. The parameters of MC-dropout experiments are similar to the work [Kirsch et al., 2019] settings, deep ensembles experiments are performed with an ensemble of 5 models.

We measure the accuracy of the model prediction on a test dataset, depending on the amount of training data obtained by different algorithms. All results are obtained as the average of the 5 runs, and the corresponding standard deviation is shown as filled error bars.

MNIST and its Variations

Experimental Setup

The first group of image experiments deals with MNIST and its extensions: MNIST [LeCun et al., 1998], Repeated MNIST (RMNIST; [Kirsch et al., 2019]), and Fashion MNIST (FMNIST; [Xiao et al., 2017]). MNIST is a standard machine learning dataset suitable for active learning that consists of handwritten digit images, including $60,000$ images from 10 classes. RMNIST is an extension of the MNIST dataset in which each image is repeated several times with a small Gaussian noise applied. FMNIST is a fashion product dataset containing $70,000$ images of 10 classes.

Model architecture for MNIST, RMNIST and FMNIST datasets is taken similar as in [Kirsch et al., 2019] for the MC-dropout uncertainty case. For experiments with deep ensembles, the same architecture was adapted to use multiple initializations of the same network to form an ensemble. Note that ensembles are more time-consuming than MC-dropout. While MC-dropout requires multiple forward passes on inference of the same network, for ensembles one needs to fully train multiple networks. Nevertheless, when using deep ensembles, better model quality can be achieved by better calibration of the resulting models [Beluch et al., 2018]. We compared the performance of the AL algorithms for both ensembles and MC-dropout uncertainty estimates on acquisition batches of 10 and 20 images on the specified datasets.

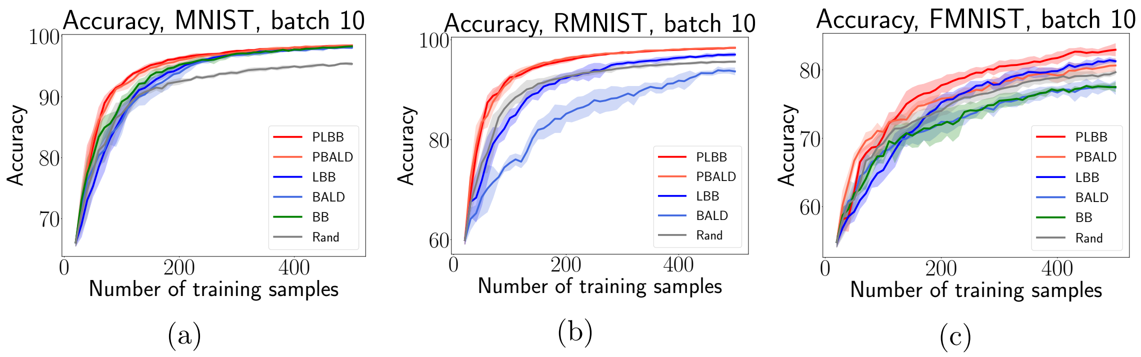


Figure 3-2: Performance comparison of AL algorithms on deep ensembles. Datasets: (a) MNIST. (b) RMNIST. (c) FMNIST. LBB (ours) shows better performance than BALD, and their randomized extensions, PLBB (ours) and PBALD, lead among other algorithms.

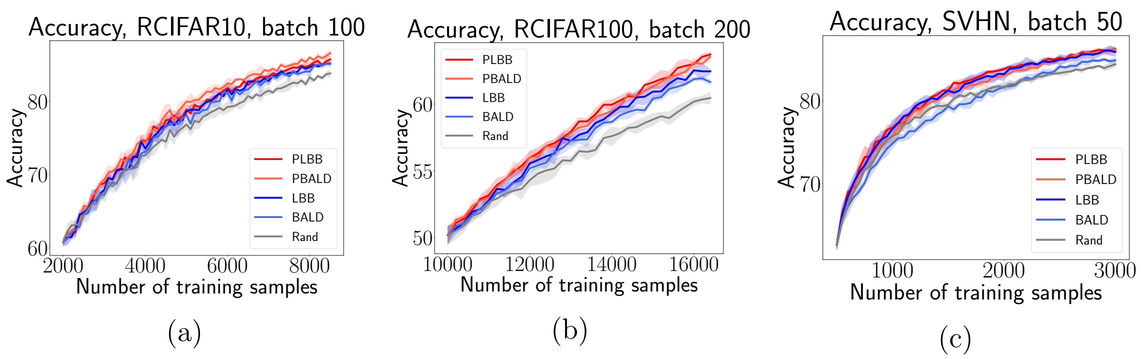


Figure 3-3: Performance comparison of AL algorithms on deep ensembles. Datasets: (a) RCIFAR-10. (b) RCIFAR-100. (c) SVHN. LBB (ours) outperforms BALD, and also shows results close to the leading methods based on power distribution, namely PLBB (ours) and PBALD.

Results

The results of the experiment on the MNIST dataset are shown in Figure 3-2a. The Large BatchBALD algorithm is slightly better than the BALD algorithm, and as a BatchBALD approximation it is quite close to the original. As for the LBB and BALD extensions, namely PLBB and PBALD, they dominate among other algorithms, even outperforming the BatchBALD algorithm in both the ensemble and MC-dropout cases.

In the RMNIST experiments, see Figure 3-2b, BALD takes many similar images that do not introduce significant diversity into the training dataset, which is reflected in a loss of quality compared to other algorithms. Large BatchBALD, on the other hand, as an approximation of BatchBALD performs much better than BALD, taking into account batch interconnections. At first, it is slightly inferior to the random baseline, but then outperforms it, starting with a few hundred elements. In turn, PLBB and PBALD, which combine informativity proportional to the LBB and BALD criterion scores, respectively, and the diversity obtained by sampling from the power distribution, are quite close to each other in their performance and outperform all other algorithms. Note that on a dataset with a large pool, like RMNIST, and on experiments with large batches, BatchBALD becomes computationally infeasible.

Regarding the results on the FMNIST dataset, the Large BatchBALD algorithm performs better than BALD. The results of its PowerLBB and PowerBALD extensions are the best among other algorithms, with PLBB significantly outperforming PBALD, see Figure 3-2c. This may be due to the fact that PLBB has the additional data diversity contained in the LBB algorithm design, while PBALD has data diversity only due to sampling-driven randomness. Both mentioned algorithms are better than BatchBALD, which in turn, together with BALD, has comparable performance and worse results among competitors. There are also results based on MC-dropout and results with larger batch sizes on the same datasets, see Appendix A.2 and A.3, respectively.

SVHN, RCIFAR-10, RCIFAR-100

CIFAR-10 and CIFAR-100 [Krizhevsky, 2009] are datasets of color images, each containing $60,000$ images consisting of 10 and 100 classes, respectively. Repeated extensions of CIFAR datasets involve repeating each of the images in the dataset multiple times with a Gaussian noise applied, namely 4 times in our case, which increases the total size of each dataset proportionally. SVHN [Netzer et al., 2011] is a Street View House Numbers dataset containing $70,000$ images.

As a model, we used ResNet-18 [He et al., 2016] architecture with an SGD optimizer with momentum $= 0.9$, weight decay $= 0.0005$, learning rate $= 0.05$. As a learning rate scheduler, we used MultiStepLR with gamma $= 0.1$, and milestones $= 25, 40$. The network was trained over 50 epochs, and the version that showed the best quality on the validation set (the validation set consists of 5K examples) was used.

The results for the SVHN dataset are presented in Figure 3-3c for a batch size 50. LBB algorithm shows superiority over the BALD algorithm and the random baseline, while the BALD algorithm is inferior to the random baseline up to 2K examples. Additional randomization significantly improves the performance of the BALD algorithm, as PBALD shows. At the same time, LBB is as good as, and in some places slightly better than, the PBALD algorithm without additional randomization. The leader among all algorithms is the randomized version of LBB, the PLBB algorithm.

The Repeated CIFAR-100 (RCIFAR-100) dataset is very challenging for the task of active learning due to the large number of classes and image repetitions, which increases the initial volume by 4 times. Moreover, such a dataset requires taking samples in large batches to get good performance in a reasonable amount of time. Note that while the results based on RCIFAR-10 (see Figure 3-3a) show only slight superiority of the LBB algorithm over the BALD algorithm, on a more complex dataset like RCIFAR-100 the differences are much more clear, see Figure 3-3b. It shows that Large BatchBALD is more successful in quality than BALD and random baseline. As for the randomized versions of LBB and BALD, namely PLBB and PBALD, they improve the results of the original, with PLBB dominating in quality among the other algorithms on this dataset. For experimental results on the mentioned datasets for larger batch sizes, see Appendix A.3.

3.2.3 Algorithm Runtime Comparison

We present numerical execution times for the considered algorithms, namely BALD, PowerBALD, BatchBALD, Large BatchBALD, and Power Large BatchBALD, see Table 3.6 which is based on MNIST dataset. Execution results are obtained with deep ensembles constructed using a small convolutional neural network. The initial pool consists of 20 images, and the unlabeled pool contains $49,880$ images. Note that these results also support the claim that LBB, being an approximation of BatchBALD, is multiple times faster. Moreover, this difference becomes even more noticeable as the batch size increases, which can give a gain of tens of times. Thus, when working with batches of hundreds of items, it is evident that the calculation of the

Large BatchBALD acquisition function is more feasible than that of the BatchBALD method.

Batch size	BALD	PowerBALD	BatchBALD	LBB (ours)	PLBB (ours)
10	5.04 ± 0.51	5.29 ± 0.49	268.9 ± 10.37	18.18 ± 1.22	20.13 ± 2.61
20	5.13 ± 0.43	5.93 ± 1.43	838.85 ± 98.22	20.06 ± 5.31	18.56 ± 2.27

Table 3.6: Algorithm runtime comparison in seconds. MNIST dataset, unlabeled pool of $49,980$ images, uncertainty estimation with deep ensembles. The proposed LBB-based approximation allows a significant reduction in computational cost compared to BatchBALD, even on batches of tens of elements.

3.3 Conclusions

To summarize, in this chapter we introduced a new active learning algorithm *Large BatchBALD* that performs an approximation of the BatchBALD method, using the BALD acquisition function and pairwise mutual information of model output components. The proposed algorithm is as efficient in avoiding taking similar objects in one batch as the original method, while it computes the acquisition function several times faster, especially in the case of large batches. Thus, this active learning algorithm balances the uncertainty and diversity of the acquired samples and significantly reduces the acquisition time compared to the original BatchBALD. The resulting method is shown to be efficient for batch AL in application to modern text and image datasets. The code to reproduce the experiments is publicly available[2].

[2] https://github.com/stat-ml/large_batch_bald

Chapter 4

Fact-Checking the Output of Large Language Models via Token-Level Uncertainty Quantification

4.1 Uncertainty Quantification

In this section, we first provide background on common UQ methods that can be used at the token level, then delve into our **Claim-Conditioned Probability (CCP)** token-level method, and finally describe how token-level uncertainties are aggregated into a claim-level score.

Autoregressive language models generate text token by token. In this work, we will operate on the level of words and without loss of generality suppose that the autoregressive distribution at each step generates the random word $X_j \sim P(\cdot \mid x_{<j})$, where $x_{<j}$ is the text generated before the word at position j. We also denote by x_j the generated word at position j and by $x_{1:j} = x_{<j} \circ x_j$ a text composed of words at positions 1 to j. For example, in the case of greedy generation, $x_j = \arg\max_x P(x \mid x_{<j})$, where x_j is the most probable realization of X_j. Let us also denote by C a set of indices of words corresponding to a particular atomic claim.

4.1.1 Claim-Level UQ Baselines

We note that for UQ to be practical, it needs to be fast. Therefore, we do not consider methods such as deep ensembles [Lakshminarayanan et al., 2017], due to their significant computational overhead.

Maximum Probability represents a basic approach to UQ, where we simply treat the probability of the most likely generation as a confidence score:

$$MP(C) = 1 - \prod\nolimits_{j \in C} P(x_j \mid x_{<j}). \tag{4.1}$$

Perplexity is a common metric used to evaluate the performance of LLMs. Lower perplexity indicates that a model's probability distribution better predicts a sample. It is computed as the average negative log probability of generated tokens that belong to the claim C:

$$Perp(C) = \exp\left(-\frac{1}{|C|}\sum\nolimits_{j \in C} \log P(x_j \mid x_{<j})\right). \tag{4.2}$$

Maximum Entropy of a token in the claim:

$$Ent(C) = \max\nolimits_{j \in C} \mathcal{H}(\cdot \mid x_{<j}), \tag{4.3}$$

where $\mathcal{H}(\cdot \mid x_{<j})$ is the entropy of the autoregressive distribution of the current token. Preliminary experiments indicated that simply getting the maximum of the token entropies in a claim noticeably outperforms other aggregation techniques like average or minimum. It is also generally a slightly better baseline than perplexity.

P(True), similar to Kadavath et al. [2022], measures the uncertainty of the claim by asking the LLM itself whether the generated claim is true or not. The confidence is the probability of the first generated token y_1 being equal to "True":

$$P(\text{True}) = 1 - P(y_1 = \text{“True”}). \tag{4.4}$$

While some work has reported that this technique outperforms other baselines, the big drawback is that one needs to run the original LLM twice.

4.1.2 Claim-Conditioned Probability

In this subsection, we propose a novel method for token- and claim-level uncertainty quantification.

Motivation and Theoretical Background

When an LLM generates an output, it faces various types of uncertainty reflected in the token distribution of the current generation step (see Figure 4-8 for an example). We identify three distinct types of uncertainty:

1) Claim type/order uncertainty: What claim to generate on the current step? For example, on the current step, an LLM might hesitate between generating a year of graduation of a person and a field of study. A different order of claims, missing claims, or different types of generated claims do not make produced text less factual. Therefore, when performing fact-checking, we should not take this type of uncertainty into account.

2) Surface form uncertainty: What synonyms or hypernyms to use when generating a claim (e.g. "art" or "painting")? Different surface forms also do not make the text less factual — they might change the style, but not the underlying meaning of the text. Therefore, this type of uncertainty is also not relevant for fact-checking.

3) Claim uncertainty: What specific piece of information to relay for a particular claim type? For example, an LLM might not be sure which field of study to generate, producing a token distribution with multiple highly-probable variants, such as "painting", "acting", "sculpture". Similarly, for the year of graduation, an LLM might produce a distribution with various potential years. This uncertainty is relevant for fact-checking, because if the model is not sure about the information it relays, there might be a high chance of a factual mistake.

Two out of the three types of uncertainty are irrelevant for fact-checking and only introduce noise into the final score. We propose a new UQ method that ignores the first two types of uncertainty and focuses only on the third one, namely **Claim-Conditioned Probability (CCP)**:

$$CCP(x_j) = P\big(\text{Meaning}(x_{1:j})|x_{<j}, \text{ClaimType}(x_{1:j})\big). \tag{4.5}$$

Here, $\text{ClaimType}(x_{1:j})$ represents a claim type of the generated sequence $x_{1:j}$ and $\text{Meaning}(x_{1:j})$ is a function that maps x_j into its meaning given the previous words in a sentence $x_{<j}$, so that various surface forms with a similar meaning for x_j are mapped to a single categorical variable.

Conditional probability can be rewritten using unconditional probabilities:

$$P\big(\text{Meaning}(x_{1:j}) \mid \text{ClaimType}(x_{1:j}), x_{<j}\big) = \frac{P(\text{Meaning}(x_{1:j}), \text{ClaimType}(x_{1:j}) \mid x_{<j})}{P(\text{ClaimType}(x_{1:j}) \mid x_{<j})}. \tag{4.6}$$

Assuming that each meaning in a word distribution can correspond to only a single claim type, the joint probability is the same as the meaning probability $P(\text{Meaning}(x_{1:j}), \text{ClaimType}(x_{1:j}) \mid x_{<j}) = P(\text{Meaning}(x_{1:j}) \mid x_{<j})$.

Meaning probability in turn sums from the probabilities of word alternatives x_j^k that correspond to the same meaning: $P(\text{Meaning}(x_{1:j}) \mid x_{<j}) = \sum_{x_j^k \in M(x_j)} P(x_j^k \mid x_{<j})$, where we say that $x_j^k \in M(x_j)$ if $\text{Meaning}(x_{1:j}) = \text{Meaning}(x_{<j} \circ x_j^k)$.

In the same way, the probability of a claim type sums from probabilities of all meanings and transitionally from probabilities of words that correspond to the particular claim type: $P\big(\text{ClaimType}(x_{1:j})\big) = \sum_{x_j^l \in CT(x_j)} P(x_j^l \mid x_{<j})$, where we denote by $x_j^l \in CT(x_j)$ an event such that $\text{ClaimType}(x_{1:j}) = \text{ClaimType}(x_{<j} \circ x_j^l)$. Therefore, equation (4.5) can be rewritten as follows:

$$CCP(x_j) = \frac{\sum_{x_j^k \in M(x_j)} P(x_j^k \mid x_{<j})}{\sum_{x_j^l \in CT(x_j)} P(x_j^l \mid x_{<j})}. \tag{4.7}$$

The meaning function and the construction of the set of words that belong to the same claim type can be implemented in various ways. We outline our approach in Section 4.1.2.

Previously-proposed UQ methods have partially accounted for some of the types of uncertainty described above. For example, semantic entropy [Kuhn et al., 2023] accounts for uncertainty in semantically-equivalent groups, which helps to alleviate the surface-form uncertainty. However, in our method, we additionally remove the impact of claim-type uncertainty.

Implementation

We implement CCP using pre-trained Natural Language Inference (NLI) model at the word level. NLI model determines whether a "hypothesis" is true (entailment), false (contradiction), or undetermined (neutral) given a "premise". We compare the original claim and the claim, in which the target word is replaced by its alternatives from the autoregressive distribution.

The distribution X_j at the position j is approximated by top-K alternatives $\{x_j^k\}_{k=1}^K$ with $x_j^1 \equiv x_j$. We replace x_j with its alternatives x_j^k and obtain new instances $x_{<j} \circ x_j^k, k = 1, \dots, K$. Each new instance is compared against the original prediction $x_{1:j} = x_{<j} \circ x_j$ using an NLI model. We define $\texttt{NLI}(x_j^k, x_j) := \texttt{NLI}(x_{<j} \circ x_j^k, x_{1:j})$, where $\texttt{NLI}(x_{<j} \circ x_j^k, x_{1:j})$ means application of the NLI model to the text fragments $x_{<j} \circ x_j^k$ and $x_{1:j}$.

The outcome of the NLI procedure is one of three labels: entail ('e'), contradict ('c'), or neutral ('n'). If the new instance entails the original prediction $\texttt{NLI}(x_j^k, x_j) =$ 'e', then we consider $x_{<j} \circ x_j^k$ has the same meaning with $x_{1:j}$ ($x_j^k \in M(x_j)$) and corresponds to the same claim type ($x_j^k \in CT(x_j)$).

If the new instance contradicts the original prediction $\texttt{NLI}(x_j^k, x_j) =$ 'c', then we consider $x_{<j} \circ x_j^k$ has a different meaning with $x_{1:j}$ ($x_j^k \notin M(x_j)$), but corresponds to the same claim type ($x_j^k \in CT(x_j)$). Otherwise, if the new instance is neutral w.r.t. the original prediction $\texttt{NLI}(x_j^k, x_j) =$ 'n', then we consider that $x_{<j} \circ x_j^k$ does not correspond to the same claim type as $x_{1:j}$ ($x_j^k \notin CT(x_j)$). Thus, equation (4.7) for CCP can be written as follows:

$$CCP_{word}(x_j) = \frac{\sum_{k:\texttt{NLI}(x_j^k, x_j) = \text{`e'}} P(x_j^k \mid x_{<j})}{\sum_{k:\texttt{NLI}(x_j^k, x_j) \in \{\text{`e'}, \text{`c'}\}} P(x_j^k \mid x_{<j})}.$$

For practical considerations, we consider that CCP for function words is always equal to 1. In our experiments, we base this determination on the stop word list from NLTK [Bird and Loper, 2004].

We note that most transformer LLMs generate sub-word tokens instead of whole words. To obtain distributions for whole words, we generate one or multiple tokens using beam search with K beams.

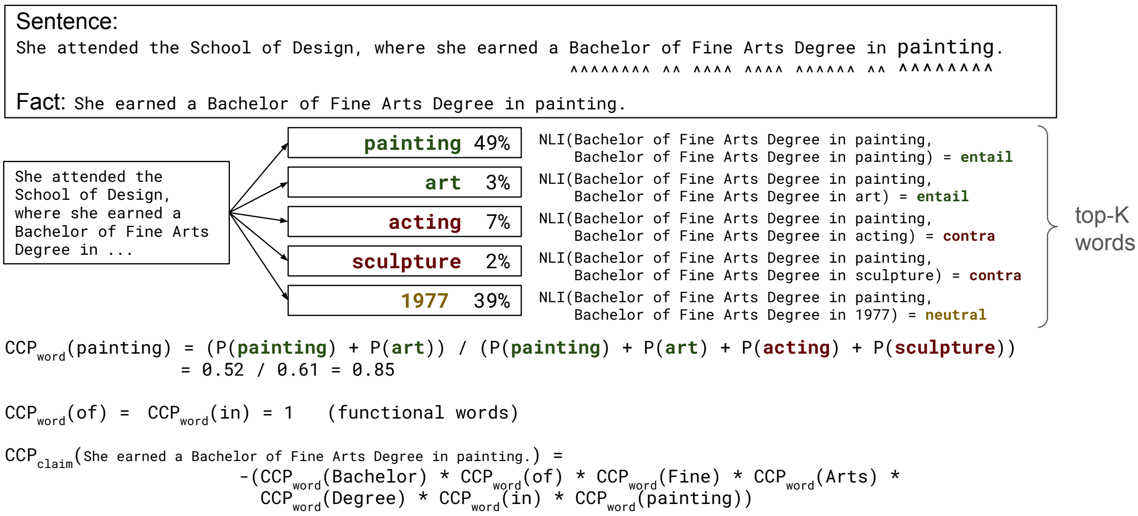


Figure 4-1: Example of CCP calculation for the word *painting* in a Vicuna 13b generation.

To obtain CCP-based claim-level uncertainty, we simply take the product of CCPs of each word from the claim C:

$$CCP_{claim}(C) = 1 - \prod\nolimits_{j \in C} CCP_{word}(x_j). \tag{4.8}$$

An example of calculating the CCP for a claim is presented in Figure 4-1. Other detailed examples of CCP calculation are available in Section 4.7.3.

4.2 Benchmark for Evaluation of Claim-Level UQ Methods

We evaluate claim-level UQ techniques and their ability to spot hallucinations on the task of generating biographies. In relevant previous work of Manakul et al. [2023], the authors generate biographies with one LLM (GPT-3), manually annotate sentences for factuality, and quantify uncertainty of a different "proxy" model. Factuality labels are then used to evaluate the quality of uncertainty scores. We argue that such an approach based on a proxy model introduces a big discrepancy between the generated text and what a proxy LLM actually wants to generate, which results in biased UQ evaluation results. To make the evaluation as close as possible to the real-world scenario, we allow unrestricted generation of biographies from LLMs.

Unrestricted generation complicates automatic evaluation of the fact-checking pipeline, because obtaining gold standard annotation requires manual annotation of all outputs from each model. Therefore, in addition to manual annotation, we annotate claims in generated texts automatically using FactScore – a fact-checking tool [Min et al., 2023], which has access to an external knowledge source. Using FactScore, enables completely automatic evaluation and allows us to scale up experiments.

We generate LLM responses in English, Chinese, Arabic, and Russian to 100 biography prompts. The typical biography prompt is *Give me a biography for* <*person name*> in different languages. The set of people was generated by asking GPT-4 to list the most famous people since 1900. The maximum generation length is set to 256 tokens. If the last sentence of the generation is unfinished (i.e. does not end with any punctuation), it is discarded. We generate responses for the following LLMs: (for English) Vicuna 13b [Zheng et al., 2024], Mistral 7b [Jiang et al., 2023], Jais 13b [Sengupta et al., 2023], and GPT-3.5-turbo [Ouyang et al., 2022]; (for Chinese) Yi 6b [Young et al., 2024]; (for Arabic) Jais 13b and GPT-4; (for Russian) Vikhr-instruct-0.2 7b [Nikolich et al., 2024].

We decompose the generated text into atomic claims using GPT-4. For each claim, we map all its words back to generated text to access the corresponding token logits. Not all claims perfectly match to the original response. For example, for Vicuna 13b around 5% of all claims do not successfully match because ChatGPT abstained to respond or outputted words not present in the original. We consider only successfully matched claims.

For English, we are able to perform annotation completely automatically: each atomic claim is classified by FactScore as supported or not supported. The underlying FactScore model is "`retrieval+ChatGPT`" with a dump of Wikipedia articles as an external knowledge source. We also manually annotated 100 claims from English biographies produced by the Vicuna 13b model, 183 claims from Arabic biographies, 146 claims from Russian biographies, and 1603 claims in Chinese. Each of the statements was checked by two annotators with access to the corresponding Wikipedia article. The final label is set to "supported" only if both annotators label it as so.

The statistics of the resulting datasets with automatically-labeled claims in English are presented in Table 4.9, and the statistics of manually annotated datasets are presented in Table 4.10. The majority of claims in the model output are correct, with 6–29% of hallucinations. The automatic pipeline for evaluation of UQ methods using FactScore is illustrated in Figure 4-7.

4.3 Fact-Checking Pipeline

The fact-checking pipeline (see Figure 4-2) starts with splitting a generated text into atomic claims, e.g. using a much smaller model fine-tuned for this particular task. For experimental evaluations in this work, we follow the FactScore approach [Min et al., 2023], where splitting is implemented via the OpenAI Chat API.

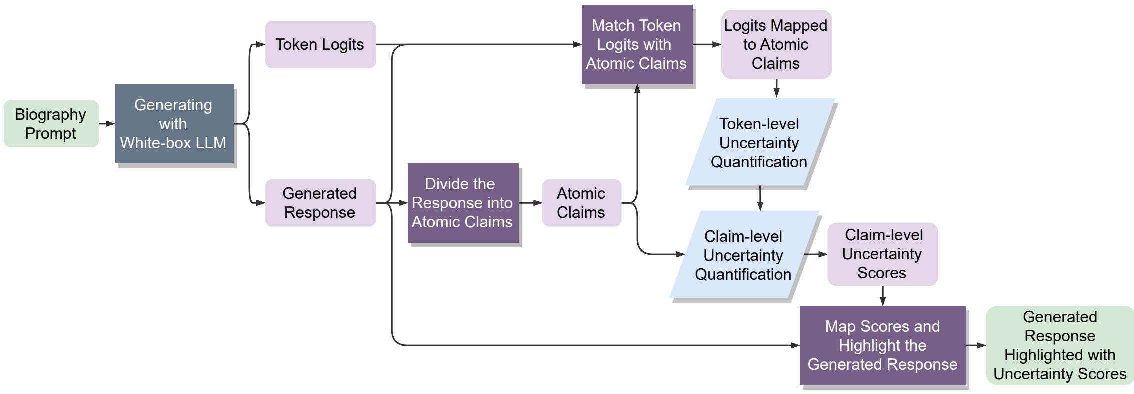


Figure 4-2: The scheme of the fact-checking pipeline based on UQ.

Each atomic claim is matched against the sequence of tokens in the original text with the corresponding probability distributions. Then we calculate token-level uncertainty scores and aggregate them into the claim-level uncertainty.

Finally, the claim-level uncertainty scores are compared against a threshold obtained on a validation set to determine whether the claim should be highlighted for the end-user as unreliable. Individual tokens can be a part of multiple atomic claims. If the token belongs to a reliable and unreliable claim at the same time, it is not highlighted. An example visualization is presented in Figure 4-3.

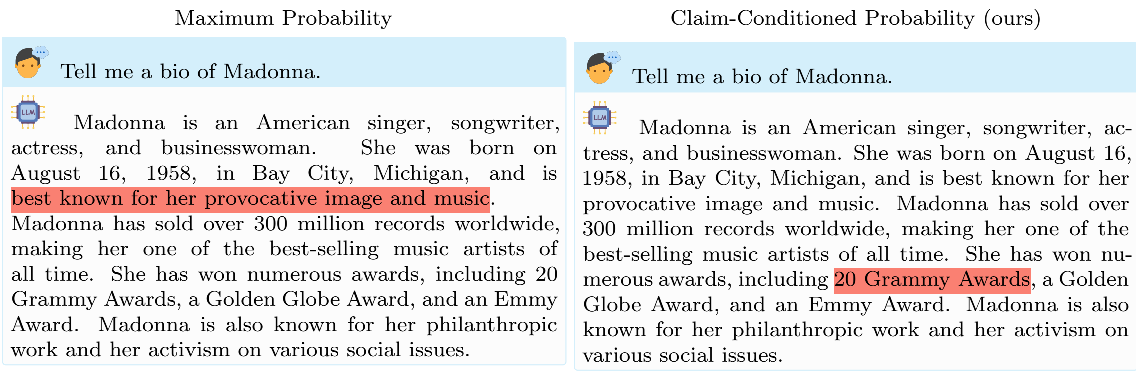


Figure 4-3: Visual comparison of our Claim-Conditioned Probability method to the Maximum Probability baseline. CCP accurately identifies the incorrectly specified number of awards (in red), whereas Maximum Probability erroneously highlights the claim that is actually correct.

4.4 Experiments

4.4.1 Experimental Setup

Fact-checking of atomic claims is framed as a binary classification task, where uncertainty scores serve as predictors of non-factuality, and FactScore or human labels serve as ground truth. The evaluation metric is ROC-AUC and PR-AUC (unsupported claims as a positive class).

For the CCP method, NLI scores are calculated using the DeBERTa-large model [He et al., 2021] fine-tuned for this task[1]. The number of alternatives used in CCP is $K = 10$, except for GPT-3.5-turbo and GPT-4, as the OpenAI API does not allow us to retrieve more than 5 alternatives from the token distribution.

4.4.2 Results for English on the FactScore Annotation

The main results of the experiments with FactScore labels for English are presented in Tables 4.1 and 4.2. The proposed CCP method outperforms all other UQ techniques for each of the considered LLMs, with only exception in the PR-AUC metric for the GPT-3.5-turbo model, where the P(True) approach exhibits the highest performance. The underperformance of CCP for GPT-3.5-turbo may be attributed to the limited

[1] https://huggingface.co/microsoft/deberta-large-mnli

Model	Mistral 7b	Vicuna 13b	Jais 13b	GPT-3.5-turbo
CCP (ours)	**0.66** ± 0.03	**0.66** ± 0.04	**0.71** ± 0.05	**0.58** ± 0.04
Maximum Prob.	0.59 ± 0.03	0.60 ± 0.05	0.64 ± 0.05	0.54 ± 0.05
Perplexity	0.58 ± 0.07	0.58 ± 0.07	0.61 ± 0.08	0.53 ± 0.06
Token Entropy	0.60 ± 0.02	0.60 ± 0.06	0.63 ± 0.06	0.53 ± 0.03
P(True)	0.53 ± 0.02	0.61 ± 0.03	0.55 ± 0.05	0.53 ± 0.04

Table 4.1: ROC-AUC of claim-level UQ methods with FactScore labels as the ground truth (English).

Model	Mistral 7b	Vicuna 13b	Jais 13b	GPT-3.5-turbo
CCP (ours)	**0.34** ± 0.05	**0.24** ± 0.04	**0.33** ± 0.07	0.14 ± 0.01
Maximum Prob.	0.26 ± 0.04	0.18 ± 0.05	0.24 ± 0.05	0.13 ± 0.02
Perplexity	0.27 ± 0.03	0.17 ± 0.04	0.21 ± 0.05	0.11 ± 0.03
Token Entropy	0.30 ± 0.06	0.18 ± 0.04	0.24 ± 0.06	0.12 ± 0.02
P(True)	0.24 ± 0.03	0.21 ± 0.06	0.19 ± 0.06	**0.17** ± 0.02

Table 4.2: PR-AUC (considering the Not Supported class as positive) of claim-level UQ methods with FactScore labels as the ground truth (English).

number of token options and their associated logits available through the OpenAI API. The best overall improvement from CPP is obtained for Jais 13b, where it outperforms the closest competitor by 0.07 ROC-AUC and 0.09 PR-AUC.

We further analyze the performance by plotting ROC-AUC as a function of the number of considered sentences from the beginning of the generated text (see Figure 4-4). We note that the quality of each method decreases as the number of considered sentences increases. This may be due to the fact that the model tends to start the response with easy-to-know and hence reliable claims and as it generates more text, it has to produce more complex and less reliable statements, which is illustrated for the Vicuna 13b model in Figure 4-9. For the majority of cases, CCP outperforms other methods, except when we consider only the first two and the first five sentences generated by GPT-3.5-turbo.

4.4.3 Multilingual Results on Manual Annotation

Multilingual results based on manual annotation are presented in Tables 4.3 and 4.4 and in Figure 4-5. Using manual annotation for English, we can also evaluate the performance of FactScore itself. The accuracy of automatic annotation is 77.2%

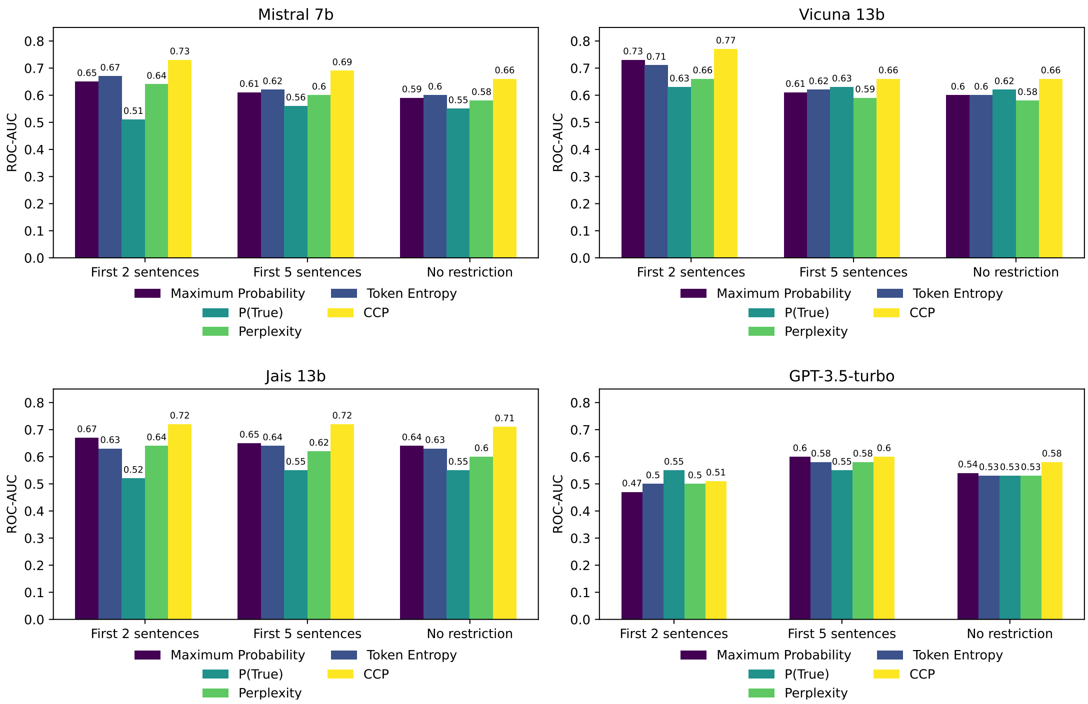


Figure 4-4: ROC-AUC of claim-level UQ methods based on FactScore labels, aggregated into bins when considering only facts from the first 2, 5, and all sentences (English).

and ROC-AUC is 0.72. A detailed analysis of FactScore mistakes is presented in Section 4.6.3.

For human annotation, the performance of all UQ methods appears to be even slightly higher than for the labels obtained using FactScore (Table 4.3). Moreover, CCP outperforms FactScore itself by 0.06 ROC-AUC. These results demonstrate that in the task of detecting LLM hallucinations, UQ techniques can be a strong alternative to fact-checking tools with an external knowledge source.

Ground-Truth / Method	Human	FactScore
CCP (ours)	**0.78**	**0.74**
Maximum Prob.	0.67	0.65
Perplexity	0.65	0.64
Token Entropy	0.69	0.65
P(True)	0.68	0.65
FactScore	0.72	–

Table 4.3: ROC-AUC of claim-level UQ methods with human annotation and FactScore annotation as the ground truth (English, Vicuna 13b model).

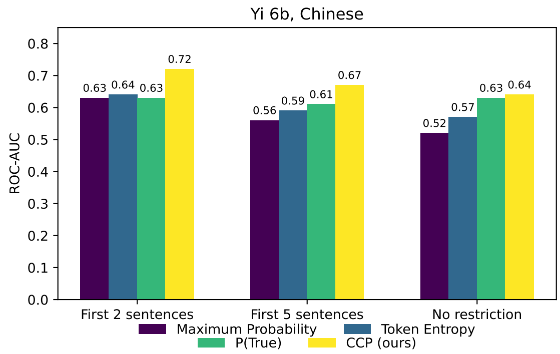


Figure 4-5: The comparison of token-level uncertainty quantification methods in terms of ROC-AUC scores, measured for Chinese dataset. The results are split into bins when considering only facts from the first 2, 5, and all sentences.

For Chinese, Arabic, and Russian, CCP also outperforms the baselines. For the Chinese Yi 6b model, from Figure 4-5, we can see that the gap between CCP and the baselines is especially significant for the several first claims. When considering more claims, CCP still clearly outperforms the Maximum Probability baseline, but the P(True) baseline substantially reduces the gap. For Arabic and Jais, CCP outperforms the closest competitor by 0.05 ROC-AUC. On GPT-4 generations for Arabic, the metrics for all methods are low. We explain this observation by a small ratio of non-factual claims in the GPT-4 output. For Russian generations with the Vikhr model, CCP confidently outperforms Maximum Probability, which is its closest competitor, by 0.05 ROC-AUC.

Model	Yi 6b, Chinese	Jais 13b, Arabic	GPT-4, Arabic	Vikhr 7b, Russian
CCP (ours)	**0.64** ± 0.03	**0.66** ± 0.02	**0.56** ± 0.05	**0.68** ± 0.04
Maximum Prob.	0.52 ± 0.03	0.59 ± 0.02	0.55 ± 0.08	0.63 ± 0.04
Perplexity	0.51 ± 0.04	0.56 ± 0.02	0.54 ± 0.08	0.58 ± 0.04
Token Entropy	0.57 ± 0.05	0.61 ± 0.02	0.48 ± 0.06	0.55 ± 0.03
P(True)	0.63 ± 0.04	0.61 ± 0.02	0.50 ± 0.06	0.58 ± 0.03

Table 4.4: ROC-AUC of claim-level UQ methods with manual annotation as the ground truth.

4.5 Ablation Studies

In this section, we analyze the influence of various CPP components on English biographies annotated with FactScore (Tables 4.5–4.8).

4.5.1 Aggregation

Method	ROC-AUC	PR-AUC
CCP_{prod}	**0.66** ± 0.03	**0.22** ± 0.05
CCP_{len}	0.65 ± 0.03	0.21 ± 0.03
CCP_{min}	0.64 ± 0.03	0.13 ± 0.04
CCP_{mean}	0.65 ± 0.03	**0.22** ± 0.04

Table 4.5: ROC-AUC and PR-AUC on Vicuna 13b generation, for different normalizations.

Aggregation of CCP_{word} for obtaining CCP_{claim}. Besides the product of probabilities, we also tried the normalized product, minimum, and average probability. All these approaches perform slightly worse than the product (see Table 4.5).

Table 4.5 presents the result of CCP on Vicuna 13b when applying 4 different kinds of aggregation:

$$CCP_{prod}(F) = 1 - \prod_{j=1}^{k} CCP_{word}(x_j),$$

$$CCP_{len}(F) = 1 - \exp\left(\frac{1}{k}\sum_{j=1}^{k} \log CCP_{word}(x_j)\right),$$

$$CCP_{min}(F) = 1 - \min_{j=1,\dots,k} CCP_{word}(x_j),$$

$$CCP_{mean}(F) = 1 - \frac{1}{k}\sum_{j=1}^{k} CCP_{word}(x_j).$$

NLI model	Parameters	ROC-AUC	PR-AUC
microsoft/deberta-large-mnli	350M	**0.66** ± 0.06	**0.24** ± 0.04
microsoft/deberta-base-mnli	86M	0.65 ± 0.06	0.23 ± 0.05
MoritzLaurer/mDeBERTa-v3-base	86M	**0.66** ± 0.06	0.21 ± 0.03
cross-encoder/nli-deberta-v3-large	350M	**0.66** ± 0.06	0.22 ± 0.04
cross-encoder/nli-deberta-v3-base	86M	0.65 ± 0.06	0.23 ± 0.05
cross-encoder/nli-deberta-v3-small	44M	0.65 ± 0.06	0.22 ± 0.05
cross-encoder/nli-deberta-v3-xsmall	22M	**0.66** ± 0.07	**0.24** ± 0.06

Table 4.6: The ROC-AUC and PR-AUC metrics of CCP on biographies generation dataset with Vicuna 13b model, when using different NLI models with specified identifiers in Huggingface library.

Method	ROC-AUC	PR-AUC
$CCP_{no\ context}$	0.64 ± 0.03	**0.24** ± 0.04
$CCP_{sent\ pref}$	0.59 ± 0.03	0.18 ± 0.04
$CCP_{claim\ pref}$	**0.66** ± 0.03	**0.24** ± 0.05

Table 4.7: ROC-AUC and PR-AUC on Vicuna 13b generation, for different contexts to input words with to an NLI model.

CCP_{prod} shows the best results and is chosen for the final CCP formula.

4.5.2 NLI Models

We investigate the influence of the specific NLI model on the performance of CCP. Table 4.6 presents the result of CCP on Vicuna 13b when using different NLI models. We selected fine-tuned NLI models from HuggingFace, encompassing a range of sizes from Microsoft's DeBERTa model [He et al., 2021], a multilingual variant [Laurer et al., 2023], and a model from SentenceTransformers [Reimers and Gurevych, 2019].

It shows that the CCP performance exhibits minimal dependence on the specific NLI model chosen. Notably, even CCP utilizing a relatively small CrossEncoder model with only 22M parameters achieves the highest performance.

4.5.3 NLI Context

We analyze what context is sufficient for NLI in CCP (Table 4.7). In addition to the standard variant in CCP, where we keep the claim that precedes the word in question, we experiment with a single target word without context and the whole sentence that precedes the target word.

Table 4.7 presents the result of CCP on Vicuna 13b when adding different contexts as the inputs of NLI model:

1. $CCP_{no\ context}$ calculates NLI using these 2 words as NLI model input;

2. $CCP_{sent\ pref}$ calculates NLI using the prefix in the sentence from model generation up to the current word and its alternative, as input to NLI model;

3. $CCP_{claim\ pref}$, uses all words in the sentence corresponding to the greedy word, which were matched to the current fact and which comes before the greedy word (see example on Figure 4-1).

The results for sentence prefix provides too wide context to the NLI model, which results in the NLI model focusing on the contexts instead of the greedy words and its alternatives, thus outputting lots of entailment classes. On the other hand, the method without context does not provide enough context to calculate NLI class with more quality. $CCP_{claim\ pref}$ shows the best results and is the one chosen for the final CCP formula.

4.5.4 Functional Words Handling

Method	ROC-AUC	PR-AUC
$CCP_{confident}$	**0.66** ± 0.03	**0.24** ± 0.05
CCP_{ignore}	0.63 ± 0.03	0.23 ± 0.06
$MP_{confident}$	0.59 ± 0.05	0.18 ± 0.04
MP_{ignore}	0.60 ± 0.05	0.19 ± 0.05

Table 4.8: ROC-AUC and PR-AUC on Vicuna 13b generation, for methods of handling functional words.

Table 4.8 presents the result of CCP and Maximum Probability on Vicuna 13b when handling functional words differently. The 2 approaches tested are:

1. $CCP_{confident}$ and $MP_{confident}$ are the versions of CCP and MP baselines, which assigns the most confident word-level score of 1.0 to words from NLTK [Bird and Loper, 2004] stopwords list, which is the same as skipping these words form the list of matched words of length k;

2. CCP_{ignore} and MP_{ignore} handles functional words similar to any other words. MP_{ignore} is the baseline used in the main section.

$CCP_{confident}$ shows better results and is the one chosen for final CCP formula. In contract, Maximum Probability performs worse if specifically handling functional words.

4.5.5 Number of Alternatives

Figure 4-6 presents the CCP ROC-AUC results on the whole model generation, depending on the number of beams n to run the method with. Taking $K = 5$ alternatives instead of $K = 10$ in CCP decreases ROC-AUC by 0.02. Figure 4-6 also demonstrates that further decreasing K reduces the performance even more. When increasing K, the performance plateaus at $K = 8$.

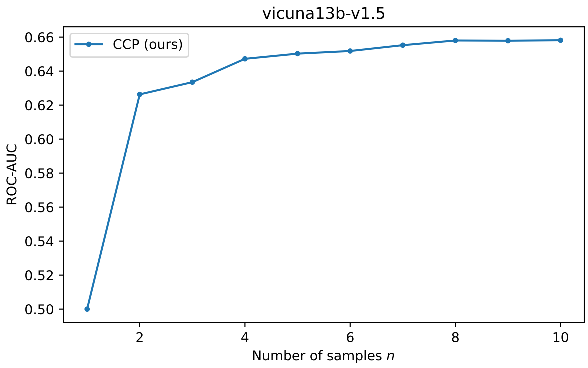


Figure 4-6: ROC-AUC between FactScore classes and the claim-level CCP method as a function of n – the number of token options in the probability distribution of the LLM to consider.

4.6 Benchmark Construction Details

4.6.1 Benchmark Construction Pipeline

Figure 4-2 presents the suggested general preparation pipeline of a factuality dataset for an arbitrary model and Fact-Checking benchmark. This pipeline was used to generate the biography dataset. Figure 4-7 presents the detailed pipeline of the biography dataset preparation, based on the general schema.

The prompt for the matching model to map atomic claims to the initial text is as follows: "`Given the fact ‘‘fact’’, identify the corresponding words in`

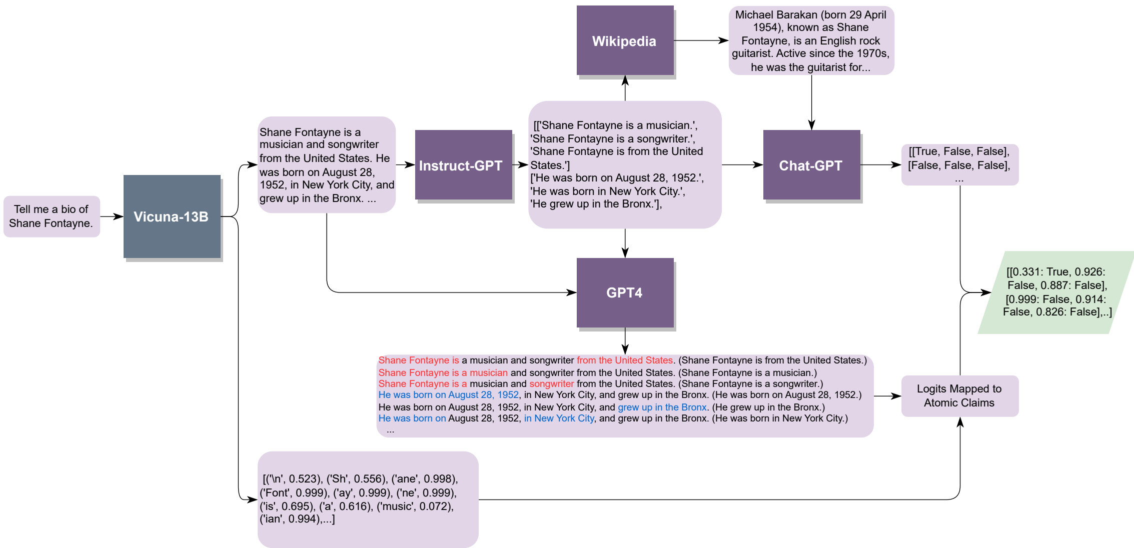


Figure 4-7: A visualization of the benchmark construction pipeline based on FactScore with an example.

`the original sentence ‘‘sent’’ that help derive this fact. Please list all words that are related to the fact, in the order they appear in the original sentence, each word separated by comma.”`. The prompt for the model that partitions the initial generated text into individual atomic claims, as well as the prompt for the classification model, are taken similar to the paper by Min et al. [2023].

4.6.2 Datasets and Statistics

Since FactScore only supports English, for Arabic, Chinese, and Russian, we generate biographies of well-known people and annotate them only manually. We also manually annotate 100 English claims generated by Vicuna 13b. The statistics for the annotated datasets are presented in Table 4.10.

For Arabic, using GPT-4, we generate 100 biographies of people randomly selected from the list of the most visited websites in Arabic Wikipedia. The used Arabic prompt is the translation of: “Tell me the biography of {person name}”. To extract claims, we prompt GPT-4 in the following way: “Convert the following biography into Arabic atomic factual claims that can be verified, one claim per line. Biography is: {biography}”. Arabic biographies and claims are translated into English using

Model	Number of claims	Supported claims
Mistral 7b	3,824	71.1%
Vicuna 13b	3,617	78.2%
Jais 13b	1,407	84.3%
GPT-3.5-turbo	3,875	89.4%

Table 4.9: The statistics of the datasets generated from 100 biographies using all tested English LLMs and annotated automatically.

Model	Number of claims	Supported claims
Vicuna 13b, English	100	70.1%
Yi 6b, Chinese	1,603	94.0%
Jais 13b, Arabic	186	73.0%
GPT-4, Arabic	200	92.5%
Vikhr 7b, Russian	146	72.6%

Table 4.10: The statistics of the datasets generated from 100 biographies using all tested English LLMs and annotated manually.

Google Translate. It is worth mentioning that almost one-third of the names in the list of person names are foreign, e.g. Donald Trump, Messi, Isaac Newton, Pope Benedict XVI, etc. On average, GPT-4 generates 20 claims from each biography, and random two claims from each biography are verified manually (total = 200 claims).

For Jais 13b experiments, we use the same prompts used for GPT-4. We notice that the biographies generated by Jais 13b are much shorter than the ones generated by GPT-4 (almost half-length). Similarly, we use GPT-4 to extract claims from the generated biographies. On average, biographies generated by Jais 13b have nine claims. Jais 13b generates empty biographies for seven names (out of 100) with response messages like: "I am sorry! I cannot provide information about {name}", or "What do you want to know exactly?". Two random claims from each biography are verified manually (total = 186 claims).

For Chinese, we first prompt ChatGPT to generate a list of 100 famous people. Then use the same way as we have done in Arabic, but change the prompt to Chinese, to generate biographies and claims. We use Yi 6b to generate texts and GPT-4 to split them into atomic claims.

For Russian, we conduct a similar approach, prompting ChatGPT to generate a list of 100 famous people and checking the result to obtain representative personalities from different areas such as science, sport, literature, art, government activity, cinematography, heroes, etc. A balanced list of famous people in different professional categories was obtained. For these people, we generate biographies using the Vikhr 7b model [Nikolich et al., 2024].

4.6.3 FactScore Annotation

Cases	FS	Human	CCP	Generated Atomic Claim	True Information from the Wikipedia article
TN	False	False	0.819	Marie Stopes died on October 20, 1958.	Marie Charlotte Carmichael Stopes (15 October 1880 – 2 October 1958) was a British author...
TN	False	False	0.999	Planck is best known for his work on the nature of light.	His fame as a physicist rests primarily on his role as the originator of quantum theory...
FN	False	True	1.0	Heisenberg was appointed as the director of the Max Planck Institute.	He then became director of the Max Planck Institute for Physics and Astrophysics from 1960 to 1970.
FN	False	True	0.716	Ray Charles incorporated elements of Latin music into his sound.	Charles reached the pinnacle of his success at Atlantic with the release of "What'd I Say", which combined gospel, jazz, blues and Latin music.
FP	True	False	0.001	Sagan was a prolific writer.	Carl Edward Sagan (November 9, 1934 – December 20, 1996) was an American astronomer and science communicator.
FP	True	False	0.141	Van Gogh was a pastor.	Van Gogh prepared for the University of Amsterdam theology entrance examination; he failed the exam ...
FP	True	False	0.185	Hawking showed an early aptitude for science.	Although known at school as "Einstein", Hawking was not initially successful academically.
TP	True	True	0.276	Tiger Woods is a professional golfer.	Eldrick Tont "Tiger" Woods (born December 30, 1975) is an American professional golfer.
TP	True	True	0.015	Miles Davis began playing the trumpet at the age of 13.	On his thirteenth birthday his father bought him a new trumpet,[17] and Davis began to play in local bands.
TP	True	True	0.001	Hitchcock directed "The Birds."	Hitchcock's other notable films include Rope (1948), Strangers on a Train (1951), ..., Birds (1963) and Marnie (1964), ...

Table 4.11: Table with examples of the FactScore automatic annotation system and all types of classification outcomes when comparing automatic annotation and manual annotation (confusion matrix elements): True Negative (TN), False Negative (FN), False Positive (FP), True Negative (TN); FS is a FactScore label, Human is a human annotation label. CCP scores are comparable to FactScore annotation and do not require an external source of information.

The statistics of English data annotated using FactScore is presented in Table 4.9. Here we give examples of the operation of the FactScore automatic markup system, see the Table 4.11. We hypothesize that the causes of FactScore errors are related to model hallucination (true information is present in the knowledge source, but the model produces incorrect information), lack of context (an excerpt from a Wikipedia article cannot capture all the details), and difficulties with information interpretation (the desired information is present in the knowledge base, but is formulated in different words or in several sentences). All these cases leave room for

Method	NLI model parameters	Runtime
MP	–	18.5 ± 0.8 sec
CCP	350M	20.1 ± 0.9 sec
CCP	22M	19.1 ± 0.8 sec

Table 4.12: The runtime of Maximum Probability and CCP methods on 100 biographies in English.

further improvement of the pipeline. We have also given the results of the proposed CCP method on selected examples. It can be seen that the results of our algorithm are similar to the annotation from FactScore, while CCP does not use external information in calculating the scores.

4.7 Additional Experimental Results

Here we show additional results related to the performance of our pipeline and CCP method.

4.7.1 Qualitative Analysis

In qualitative analysis of uncertainty scores for various generations and models, we note that the maximal probability baseline produces a lot more false positives than CCP. This happens because CCP ignores some types of uncertainty, focusing only on the claim uncertainty. In some cases, CCP also finds false claims overlooked by other methods because ignoring certain types of uncertainty also allows us to reduce the cut-off threshold used to mark claims. An example where we compare CCP with maximal probability is presented in Figure 4-12, and more examples can be found in Section 4.7.5.

4.7.2 Computational Efficiency

To demonstrate the computational efficiency of CCP, we compare it to the fastest UQ method – Maximum Probability. Experiments were conducted using a dataset of 100 biographies and Mistral 7b [Jiang et al., 2023]. To ensure a fair comparison, we focus solely on the runtime of generating biographies and calculating the respective uncertainty scores for each claim. Time spent on claim extraction and matching is excluded. The experiments utilized two 32GB V100 GPUs. Each biography was processed in a single batch.

MP does not introduce notable overhead over the generation process, as it only aggregates produced logits. CCP involves running an NLI model for each of 10 token candidates per token position, which introduces some overhead.

Table 4.12 presents the runtime comparison. We see that NLI does not make a substantial impact. Using the default microsoft/deberta-large-mnli model (350M parameters) results in 8% increase of the runtime compared to MP. Using the smaller cross-encoder/nli-deberta-v3-xsmall model (22M parameters), which achieves comparable UQ performance, reduces the computational overhead to only 3%.

4.7.3 Example of CCP Calculation

Figure 4-8 demonstrates an example of the LLM generation process and CCP calculation process. CCP quantifies the Claim Uncertainty, not taking into account the Claim order and Surface form uncertainties. As a result, CCP produces better uncertainty scores than Maximum Probability.

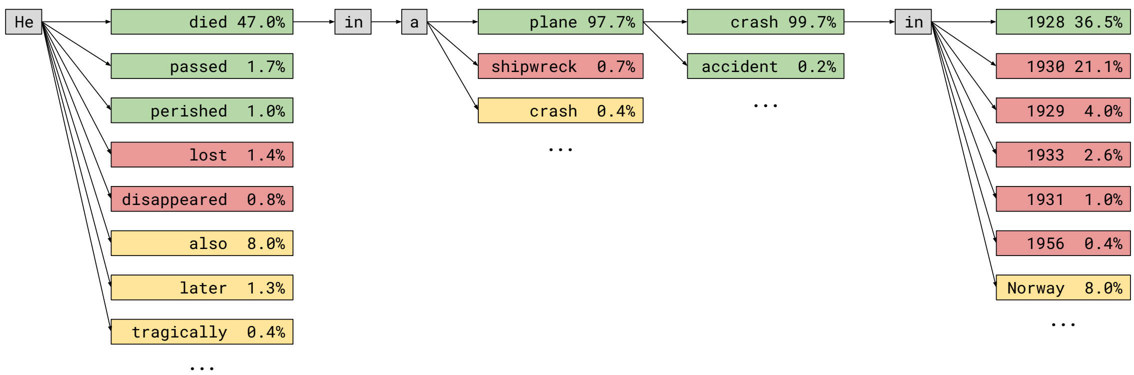


Figure 4-8: Example of the Vicuna 13b generation process and CCP calculation process part. The words from the greedy-generated sentence are presented sequentially on the top, each non-functional word is supplemented with its alternatives and autoregressive generation probabilities. Words with probability less than 0.1% are omitted.
Green-colored words indicate entailment to the greedy generated word, red color indicates contradiction, and yellow color indicates neutral NLI class.
On the last position, CCP successfully distinguishes `Norway` from other year-related words, and does not consider its probability in the final formula.

4.7.4 Supported claims

In the Figure 4-9 we show the dependence of the percentage of supported claims in the generated LLM response (in this case for Vicuna 13b) as a function of its length. As we can see, the LLM produces wrong claims when the generation length increases, which may be due to the generation of additional facts.

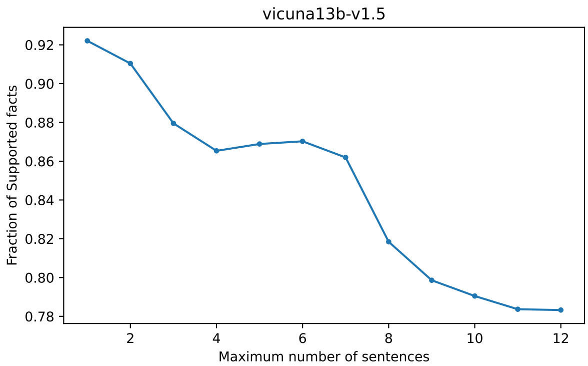


Figure 4-9: Percentage of supported claims, as a function of the number of sentences to restrict generation to (English, Vicuna 13b).

4.7.5 Examples of Results Obtained with CCP and the Maximal Probability Baseline

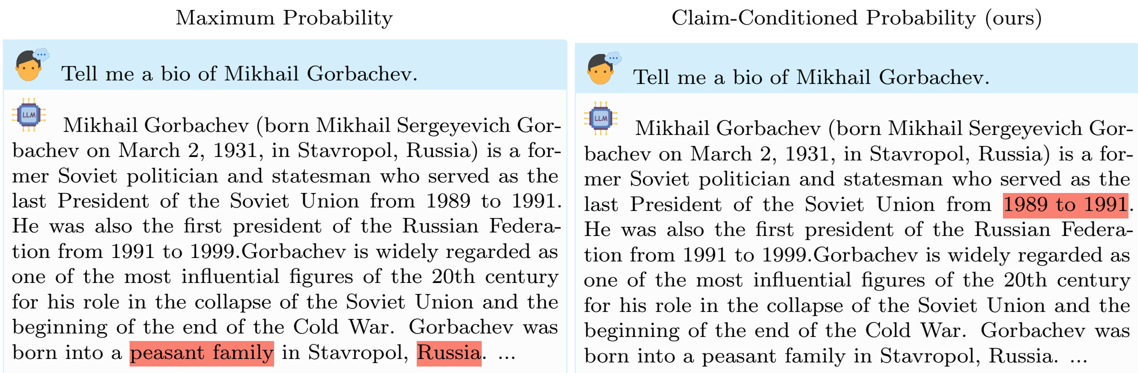


Figure 4-10: The Maximum Probability baseline vs Claim-Conditioned Probability (ours).

Figures 4-10, 4-11 and 4-12 show examples of the Claim-Conditioned Probability method compared with Maximum Probability. On all examples, our CCP method

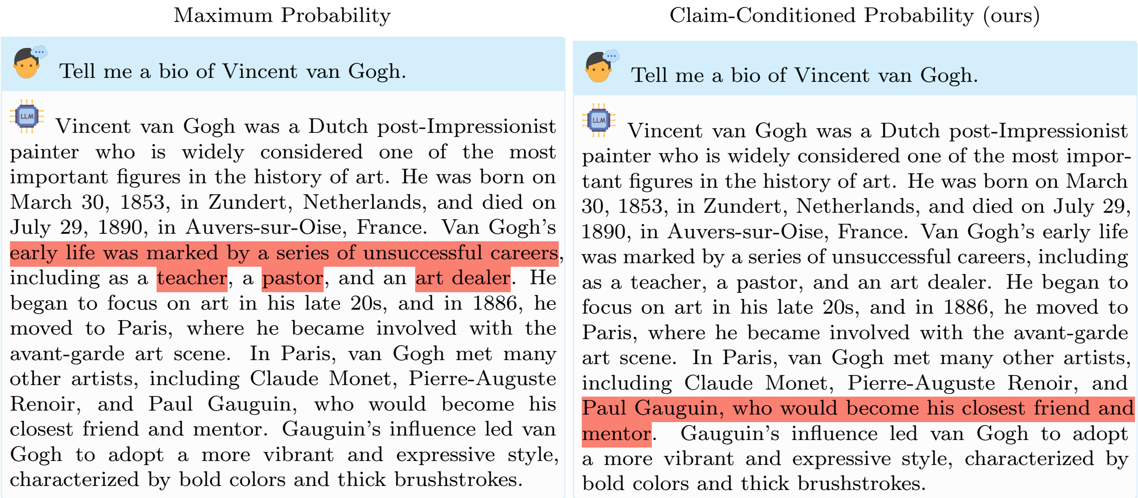


Figure 4-11: The Maximum Probability baseline vs Claim-Conditioned Probability (ours).

correctly detects the wrong facts, whereas Maximum Probability mistakenly highlights correct facts.

4.8 Conclusions

In this chapter, we presented a novel approach to fact-checking and hallucination detection based on token-level uncertainty quantification. According to human evaluation, our approach is competitive with FactScore, a fact-checking tool that leverages an external knowledge source: we achieve similar or better results with access to only LLM outputs.

We proposed a computationally efficient token-level and claim-level UQ method, Claim Conditioned Probability[2] that outperforms a number of baselines in fact-checking. In this method, we post-process the word distribution to mitigate the impact of uncertainty related to the variability of surface forms and uncertainty about what claim type to generate on the current step. In the constructed benchmark, where we detect hallucinations in biographies, CCP outperforms other methods for seven LLMs, including GPT-3.5-turbo and GPT-4, and four languages. We also demonstrate that computational overhead of CCP might be as low as 3% of the LLM

[2] https://github.com/IINemo/lm-polygraph

Maximum Probability

Tell me a bio of Bill Gates.

Bill Gates, born William Henry Gates III on October 28, 1955, is an American entrepreneur, software developer, and philanthropist. He is best known as the co-founder of Microsoft Corporation, one of the world's largest and most successful software companies. His net worth is estimated to be around $131.7 billion and is consistently ranked among the world's wealthiest people. Gates grew up in Seattle, Washington, with his parents, an attorney and a schoolteacher, and two sisters. He showed an early aptitude for mathematics and computer programming and attended Lakeside School, a private preparatory school in Seattle where he met and befriended his future business partner, Paul Allen. In 1973, Gates enrolled at Harvard University, but he soon found himself uninterested in his classes and spent much of his time on programming projects. He eventually dropped out of Harvard to focus on developing Microsoft. Gates and Allen started Microsoft in 1975, initially providing software for the Altair 8800, an early personal computer.

Claim-Conditioned Probability (ours)

Tell me a bio of Bill Gates.

Bill Gates, born William Henry Gates III on October 28, 1955, is an American entrepreneur, software developer, and philanthropist. He is best known as the co-founder of Microsoft Corporation, one of the world's largest and most successful software companies. His net worth is estimated to be around $131.7 billion and is consistently ranked among the world's wealthiest people. Gates grew up in Seattle, Washington, with his parents, an attorney and a schoolteacher, and two sisters. He showed an early aptitude for mathematics and computer programming and attended Lakeside School, a private preparatory school in Seattle where he met and befriended his future business partner, Paul Allen. In 1973, Gates enrolled at Harvard University, but he soon found himself uninterested in his classes and spent much of his time on programming projects. He eventually dropped out of Harvard to focus on developing Microsoft. Gates and Allen started Microsoft in 1975, initially providing software for the Altair 8800, an early personal computer.

Figure 4-12: The Maximum Probability baseline vs Claim-Conditioned Probability (ours).

inference runtime.

Chapter 5

Factcheck-Bench: Fine-Grained Evaluation Benchmark for Automatic Fact-checkers

5.1 Fact-Checking Pipeline

To evaluate all possible intermediate results of automatic fact-checking systems, the framework should cover as many fine-grained steps as possible. Based on our review of most prior systems and datasets, we frame the automated detection and correction of factual errors for outputs of LLMs into eight subtasks: 1) decomposition; 2) decontextualisation; 3) checkworthiness identification; 4) evidence retrieval and collection; 5) stance detection; 6) correction determination; 7) claim correction and 8) final response revision. Figure 5-1 presents the overview of the whole procedure, coupled with an example flowing through each subtask.

1) Decompose. Given a response R generated by a LLM, it is infeasible to fact-check the whole document at once, especially when it is long. The first step is to break down R into context-independent sentences. Then, we decompose each of the sentences into a set of atomic statements, as illustrated on the left side of Figure 5-1, with no information lost or distorted in this process [Fan et al., 2020, Wright et al., 2022, Chen et al., 2022]. Decomposed statements should be checkable independently

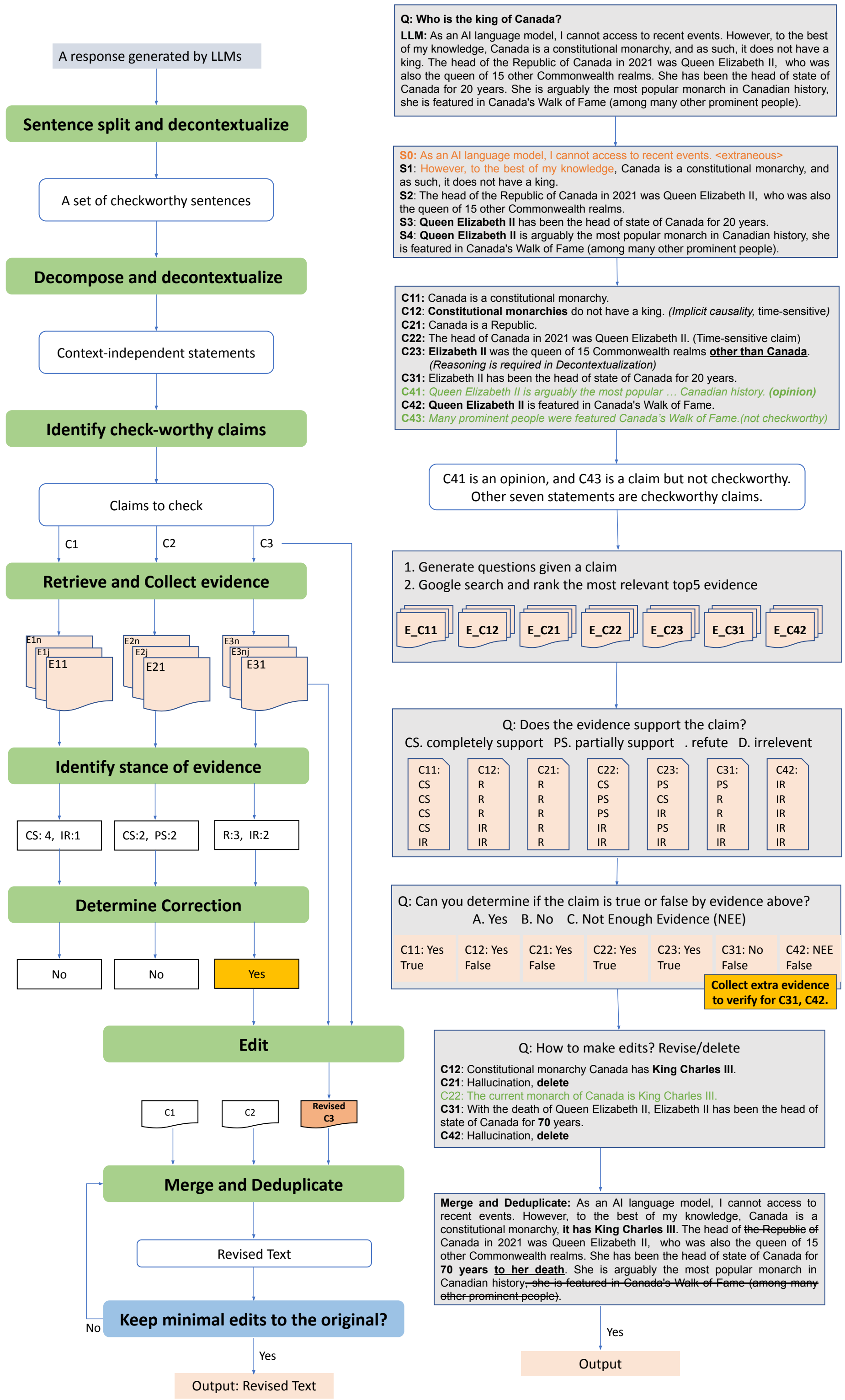


Figure 5-1: **Left:** Fact-checking pipeline for a response generated by LLMs. **Right:** An example workflow.

without preceding and following context.[1] See Table 5.1 for detailed prompt.

2) Decontextualize. Sentences in a response might be context-dependent, with discourse and coreference relations existing between statements [Choi et al., 2021]. For example, it is invalid to check the statement *It does not have a king* before replacing "It" with "Canada" or "Constitutional monarchy" (see Figure 5-1). In addition to coreference relation, for the sentence S2, it is not reasonable to check the claim *Queen Elizabeth II is also the queen of 15 other Commonwealth realms.* Instead, the claim should be reframed to *Queen Elizabeth II is the queen of 16 Commonwealth realms (including Canada)* or *Queen Elizabeth II was the queen of 15 Commonwealth realms other than Canada.*

The concept of "context-independent" is straightforward, while the notion of "atomic" is subjective and ambiguous. This poses challenges: how to determine the granularity of *an atomic claim*? when and where to break down a response? For example, S1: *Canada is a constitutional monarchy, and as such, it does not have a king*, can be fact-checked as one statement, or be decomposed into two claims: *Canada is a constitutional monarchy* and *Canada does not have a king.* In our work, we first split a document into sentences, and then from sentence to claims, with each claim containing only one property or fact to verdict.

3) Identify Checkworthy Claims. Not all statements in a response require fact-checking, such as subjective opinions and actual commonsense, as obvious as *sun rises from the east.* Each statement in this framework will be identified whether it is checkworthy or not. However, checkworthiness is subjective to determine. Hassan et al. [2015] defined checkworthy claims as those for which the general public would be interested in knowing the truth. In the context of fact-checking LLMs outputs, we assume users who ask LLMs questions are interested in knowing the truth of all factual claims in the corresponding answer.

We specifically classify a statement into four categories: factual claim, opinion, not a claim (e.g. questions, exclamations, imperatives), and others (e.g. *As a language*

[1] Statements are assumed to be checkable if relevant documents exist in publicly-available data sources.

model, I cannot...). Afterward, a set of check-worthy factual claims needs to be verified by retrieving and collecting evidence.

Note that for checkworthiness, we not only take account of objective fact against subjective judgement, other aspects such as the role (importance) of the claim to the response is also a crucial criterion for its checkworthiness. For example, the sentence S1 needs more attention than the last sentence S4 in Figure 5-1. We label the importance level of both decomposed sentences and claims by labels: *most important*, *intermediate*, and *less important*.

4) Retrieve and Collect Evidence. Evidence can be retrieved by a search engine like Google, or deep retrieval from a closed document collection such as Wikipedia, or using the parametric knowledge of a LLM. Search queries can be questions covering different aspects of the claim, entities in the claim, or even the claim itself [Dai et al., 2023]. We used Google search, considering the quality and coverage.

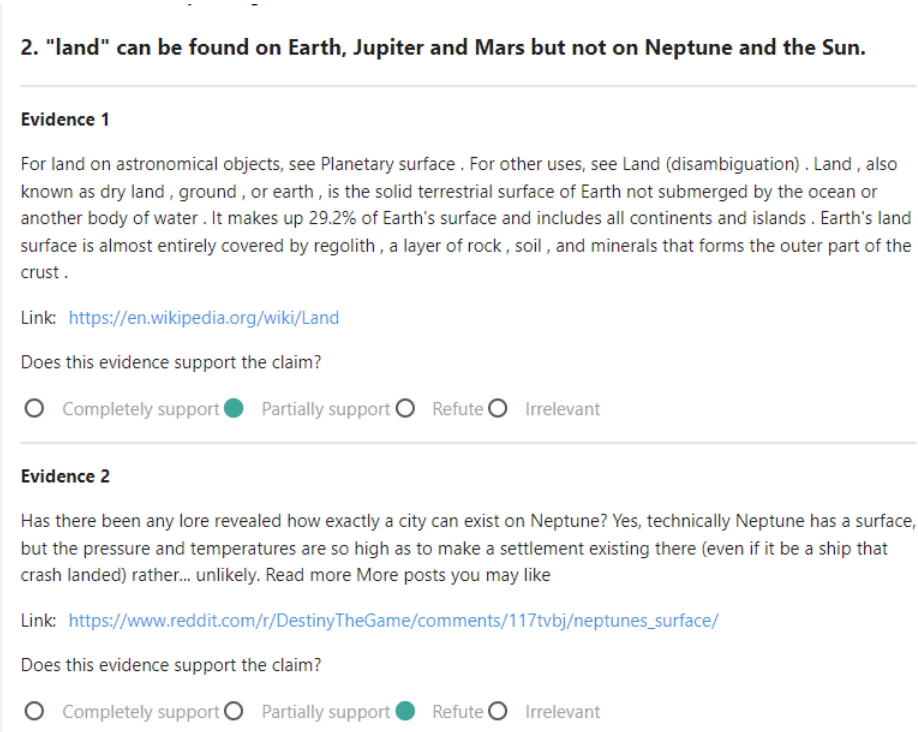
5) Identify Stance of Evidence. With retrieved evidence for a claim, how to identify the stance of the evidence against the claim. RARR [Gao et al., 2023] achieved this by assessing whether answers depending on the evidence and the claim are the same or not, given a query. If they are same, then the evidence supports the claim, otherwise refutes it. Previous work also employs Natural Language Inference (NLI) model to classify whether the claim can be entailed by evidence, or is controversial against evidence, or is irrelevant [Martín et al., 2022, Jullien et al., 2023].

However, some evidence may neither refute nor fully support a claim. This mainly results from the fact that it is always possible that the evidence supports part of the claim. For example, for the claim *Elon Musk is the founder, CEO, and chief engineer of SpaceX*, evidence *Elon Musk is the CEO of SpaceX, Tesla, and Twitter* falls into this category. The evidence supports the factual statement of *Elon Musk is the CEO of SpaceX*, but it does not provide information regarding whether *Elon Musk is the founder and chief engineer of SpaceX*.

Therefore, we incorporate *partially support* in addition to *support*, *refute* and *irrelevant* [Kamoi et al., 2023]. Concretely, *support* means that the evidence entails the claim. *Partial support* refers to the scenario where part of the information presented in a claim appears in the evidence. *Refute* means that the evidence mentions the same event as the claim, but a clear opposite fact contrasting to a part or the whole facts presented in a claim. *Irrelevant* refers to the situation that the evidence does not mention anything about the fact described in the claim, such that it neither supports nor refutes the claim.

Sometimes, it is ambiguous to distinguish between *refute* and *irrelevant*. We highlight that the evidence shows a clear opposite fact under *refute* stance, while the evidence does not include relevant facts mentioned in the claim under *irrelevant*.

6) Determine Correction. Given a claim, there will be more than one piece of related evidence. Most of the time, they hold consistent stances except for irrelevance, but sometimes, some support, some partially support while some refute (see Figure 5-2). How to aggregate conflicting stances and further decide how to make corrections to the claim is an open question. In practice, when evidence paragraphs conflict with each other, we will take the reliability of the evidence source into consideration and, meanwhile, retrieve extra information to judge which one is more dependable.



2. "land" can be found on Earth, Jupiter and Mars but not on Neptune and the Sun.

Evidence 1

For land on astronomical objects, see Planetary surface . For other uses, see Land (disambiguation) . Land , also known as dry land , ground , or earth , is the solid terrestrial surface of Earth not submerged by the ocean or another body of water . It makes up 29.2% of Earth's surface and includes all continents and islands . Earth's land surface is almost entirely covered by regolith , a layer of rock , soil , and minerals that forms the outer part of the crust .

Link: https://en.wikipedia.org/wiki/Land

Does this evidence support the claim?

Completely support | Partially support | Refute | Irrelevant

Evidence 2

Has there been any lore revealed how exactly a city can exist on Neptune? Yes, technically Neptune has a surface, but the pressure and temperatures are so high as to make a settlement existing there (even if it be a ship that crash landed) rather... unlikely. Read more More posts you may like

Link: https://www.reddit.com/r/DestinyTheGame/comments/117tvbj/neptunes_surface/

Does this evidence support the claim?

Completely support | Partially support | Refute | Irrelevant

Figure 5-2: A claim with conflicting stance evidence: *partially support* and *refute*.

A label often used is *not-enough-evidence* if there is insufficient information to make the veracity prediction, e.g., all retrieved evidence is irrelevant or intricate contradictory evidence [Atanasova et al., 2022]. So we set three labels in terms of factuality: true, false, and not-enough-evidence.

7) Edit Claims. With the principle that revised claims should preserve the text's original intent and style. Without adding or changing unnecessary additional information, we include edit operations: delete the whole claim, replace X with Y, and delete X, where X and Y are meta information in a claim. For other related works, see Thorne and Vlachos [2021], Schuster et al. [2021].

8) Revise Response. After revision, we merge statements in the original order, including non-checkworthy statements, true claims, and revised claims. Finally, we delete reduplicative content if applicable, outputting a correct and fluent response.

Our annotation framework splits the fact-checking pipeline into eight steps, more fine-grained than existing systems. This intends to incorporate all subtasks and attributes relevant to automated fact-checkers, so that the comprehensive labels can cover evaluations of a wide array of unit modules within fact-checking systems. Practical implementation of fact-checkers can merge some steps, e.g., decomposition and decontextualization into one, and evaluate the results of context-independent claims. FELM [Chen et al., 2023] annotated sentence-level *true or false* labels without correction and showed that factual error detection performance tends to be improved when utilizing claim-based segmentation methods compared with sentences. Therefore, we annotate a claim-based document-level fact-checking dataset in Section 5.2.

5.2 Benchmark Construction

We annotate a dataset serving as a benchmark evaluating the effectiveness of approaches for LLM fact-checking subtasks or the whole pipeline, and few-shot demonstration examples.

5.2.1 Data Collection

What kind of LLM generations are we most concerned about? In the context of detecting and correcting factual errors, we focus on generations in which the majority of statements are objective facts rather than subjective opinions whose veracity is not checkable. Additionally, we are more interested in questions where LLMs are prone to hallucinate or produce factual errors in responses. The whole annotation process is extremely time-consuming, about 15–30 minutes for an instance, even if with the annotation tool to ease the procedure. This requests us to sample examples that highly satisfy two criteria — **fact-intensive** and **factually-false**.

Twitter posts and in-house brainstorming. We first collect (question, response) pairs from ChatGPT/GPT-4 failures found on social media, in Web articles, and in related papers.[2] The query should satisfy the criteria that the corresponding response must have factual errors, rather than failures regarding reasoning, math, coding, bias, and so on; (query, response) also should be independent of a dialog. This results in 23 examples. We additionally brainstorm a spectrum of questions depending on individual usage experience of ChatGPT and then select 22 questions whose responses contain factually-false content by manually verifying suspicious facts.

Dolly-15k. We further employ data from Dolly-15k, which is brainstormed by thousands of *Databricks* employees. It consists of 15,011 examples, with eight categories ranging from closed, open, and general QA, to creative writing, brainstorming, information extraction, summarization and classification.[3] Since we pay attention to open-domain generations and responses with more factual statements, closed and open-question answering pairs are chosen to be the database.

We first generate ChatGPT responses for 1,773 closed QA pairs without using context information (a paragraph extracted from Wikipedia relevant to the question), and 3,700 open QA pairs. After filtering questions that cannot be answered without context as well as questions ChatGPT does not answer, we further filtered responses

[2] `https://github.com/giuven95/chatgpt-failures`

[3] Its use is subject to the *CC BY-SA 3.0* license.

with fewer than 200 characters. Taking human answers as the gold reference, we assume that if machine generations are semantically far from human answers, they may contain false information. So we keep the examples where the cosine similarity ≤ 0.5 between human answer and machine response based on SimCSE sentence embedding. Finally, we select 563 examples from closed QA and 528 from open QA, thus 1,136 (question, response) pairs in total with 45 from the first source.

5.2.2 Data Selection

The whole annotation process is extremely time-consuming, about 15-30 minutes, for an instance, even if with intermediate results from automatic methods to ease the procedure. This requests us to cherry-pick examples that highly satisfy two criteria — fact-intensive and factually-false. Therefore, we leverage FActScore to estimate the percentage of incorrect claims in a response and to filter cases with the following four steps.

Sentence split and atomic claims breaking-down. We first split a document into sentences using the NLTK tokenizer. The most straightforward way is to prompt ChatGPT to split a sentence into claims, given the response as context. However, processing sentences one by one consumes both time and API tokens.

Therefore, given the whole response as the context and the first sentence of the response, we ask ChatGPT to break the input sentence into independent atomic claims, and also continue the decomposition of the next sentence of the response (see the prompt in Section 5.3). Specifically, ChatGPT is given three demonstration examples, so that it can follow the instruction to first break down the input sentence into atomic claims, and then sequentially find the next sentence and make the splits. Over 90% examples follow the instruction, breaking down the whole response. 105 out of 1,136 examples only decompose the first sentence, on which we process sentence by sentence based on the NLTK sentence splits.

Another reason why we ask ChatGPT to re-split the response into single sentences is that we observed that some sentences are incorrectly split into smaller units by NLTK, such as decomposing a paper reference into a set of metadata, while ChatGPT can remain the citation reference as a whole.[4] A weakness of ChatGPT outputs compared with traditional models is that it is sometimes non-trivial to parse the results from the text-free responses when ChatGPT does not follow the output format as the instruction. In such cases, we have to process examples specifically.

One may argue that why not directly decompose the whole response into atomic claims, but through single sentences and then to atomic claims? There are two reasons:

- *Avoid distortion:* atomic claims decomposed and decontextualized from a whole response by models such as ChatGPT tend to either lose or hallucinate information compared to the original response, while the quality of atomic claims of a single sentence is much better;
- *Improve annotation quality:* sentences as the intermediate state, it is easier for annotators to go through 1-5 atomic claims for a sentence as one annotation unit, instead of >5 claims for a whole response (particularly long documents), which helps annotators to pay attention and improve the annotation quality.

5.3 Prompt to Generate Atomic Claims

Evidence collection for atomic claims. Given an atomic claim, following Gao et al. [2023], we first prompt ChatGPT to generate search queries for the claim, and then Google Search is used to get relevant web pages. We further split the retrieved documents into passages by sliding windows, and use a re-ranker combining lexical and semantic similarity to identify the most relevant passages for the given query, in which Sentence-BERT [Reimers and Gurevych, 2019] serves for semantic embeddings.[5] Finally, we aggregate evidence for all queries and select the top-5 evidences per atomic claim, similar to approaches presented by Laban et al. [2022] and Schuster et al. [2022].

[4] In our dataset, we prioritize sentence splits by ChatGPT, using NLTK results for unsuccessfully-parsed instances. The prompt is initialized with the first sentence split by NLTK.

[5] *cross-encoder/ms-marco-MiniLM-L-6-v2*: `https://huggingface.co/cross-encoder/ms-marco-MiniLM-L-6-v2`

Table 5.1: **Prompt** used to decompose and decontextualize a sentence into a set of independent atomic claims. We use three examples as demonstrations to elicit ChatGPT follow the instructions, break the response into sentences, as well as break a sentence into atomic claims. One of these examples is shown below.

Field	Content
Prompt	Depending the context, please breakdown the following sentence into independent facts. **Context:** The United States has had two black presidents: Barack Obama, who served two terms from 2009 to 2017, and Donald Trump, who served one term from 2017 to 2021. Obama was the first black president in the history of the United States. He was born in Honolulu, Hawaii, to a mother from Kansas and a father from Kenya. Trump was the second black president. He was born in New York City and previously served as a businessman and reality television personality. **The sentence is:** The United States has had two black presidents: Barack Obama, who served two terms from 2009 to 2017, and Donald Trump, who served one term from 2017 to 2021. **Atomic facts for this sentence are:** ["The United States has had two black presidents: Barack Obama and Donald Trump.", "Black president Barack Obama served two terms from 2009 to 2017.", "Black president Donald Trump served one term from 2017 to 2021."] **The sentence is:** Obama was the first black president in the history of the United States. **Atomic facts for this sentence are:** ["Obama was the first black president in the history of the United States."] **The sentence is:** He was born in Honolulu, Hawaii, to a mother from Kansas and a father from Kenya. **Atomic facts for this sentence are:** ["Barack Obama was born in Honolulu, Hawaii.", "Barack Obama mother was from Kansas.", "Barack Obama father was from Kenya."] **The sentence is:** Trump was the second black president. **Atomic facts for this sentence are:** ["Trump was the second black president."] **The sentence is:** He was born in New York City and previously served as a businessman and reality television personality. **Atomic facts for this sentence are:** ["Donald Trump was born in New York City.", "Donald Trump previously served as a businessman", "Donald Trump previously served as a reality television personality."]

FActScore calculation. FActScore [Min et al., 2023] is an automatic metric for fine-grained evaluation of the factuality of long-form generations. Given a generation, FActScore is calculated as the percentage of atomic claims within the generation that are supported by a knowledge source. For verifying the claim, we use the gathered evidences as input, along with the claim, and an instruction-tuned model as the verifier.

Example selection. Figure 5-3 shows the FActScore distribution of three component sources and the whole data set. We keep all 45 pairs from the first source, and Dolly examples whose FActScore are less than 0.2, resulting in 33 closed question-answering pairs and 37 open questions, in total of 115 examples. We remove a similar question (7 and 13 are similar), and four questions where the LLM did not provide helpful answers due to its inherent disability to access real-time data. For example, the LLM cannot browse the internet and does not have access to the latest information (*"which paper got the most citations in the question generation area?"* and *"which large language model contains the most parameters?"*), or up-to-date data and event-specific details ("*who was the general chair of COLING 2023*"), or individual information ("*what are the awards that Prof. William Yang Wang have?*"). We eventually annotated 110 examples in our first annotation stage, and more cases would be annotated in the next stage.

One may concern that selecting questions where LLMs are prone to hallucinate or produce factual errors in the response may introduce bias into the dataset towards hallucinating errors. However, our benchmark is intended not to evaluate the factuality ability of the LLM, but to measure the verification accuracy of automatic fact-checkers. Hence, we chose questions that are selectively hard prompts, so that the LLM responses contain challenging claims which distinguish the performance of different automatic fact-checking approaches. We argue that if most claims in the benchmark are true and easy to verify, the benchmark would not be of much use to evaluate fact-checker's performance.

Another issue is how Google's use of LLM in their queries might impact the quality of the retrieved and gathered evidence. In our view, we expect to have access

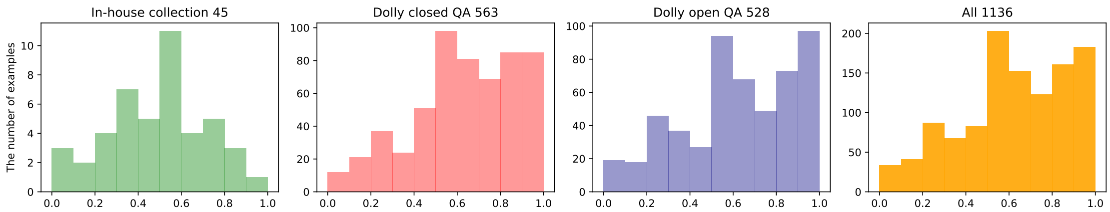


Figure 5-3: FActScore distribution for three component sources and their combination.

to the web articles from the traditional web search engine. It would be worthwhile to see whether the LLM answers to queries can be directly used as evidence for verification, given concerns of hallucinations even in retrieval-augmented generation settings.

5.3.1 LLM Responses Annotation

As many studies illustrated, annotating a LLM factuality benchmark is a highly challenging and time-consuming task [Chen et al., 2023, Li et al., 2023]. Our preliminary trials, with manual annotation of labels of all subtasks, empirically confirm the pain.

Preliminary Trial. Based on the annotation guideline (see Section 5.4), we first conduct an in-house annotation for ten examples, each example has two annotators. We annotate the whole process for all steps and manually type results into a *json* file as the pre-defined format. This attempt exposes three issues.

First, it is extremely time-consuming. It takes more than four hours for a fully-focused annotator to annotate a document of ∼400 words with about 20 sentences, in which evidence collection takes the most time and effort, particularly for topics with which the annotator is not familiar. Second, it is ineffective to extract relevant evidence passages by human eyes and basic string matching from retrieved Google search documents. This not only takes time but, most importantly, takes the risk of missing the most relevant evidence due to limited traversal. It is impractical for humans to go through all relevant web articles and select the most semantically-relevant and reliable ones in a limited time. Humans are good at judging or making decisions, while machines are good at traversing. Lastly, it is hard to reach a high agreement between annotators, especially for subtasks of decomposition, evidence collection, and stance identification.

Taking the factors mentioned above into consideration, we design and build an annotation tool to support the efficient construction of the LLM factuality benchmark. Annotators can edit and assign labels based on intermediate outputs of automatic methods, click buttons instead of typing to copy-paste text, select, and download annotated results. Decomposition, Decontextualization and Check-worthiness detection are presented in Figure 5-4, Evidence stance identification and Claim correction are presented in Figure 5-5, and Claim Merge and Deduplication are presented in Figure 5-6. [6]

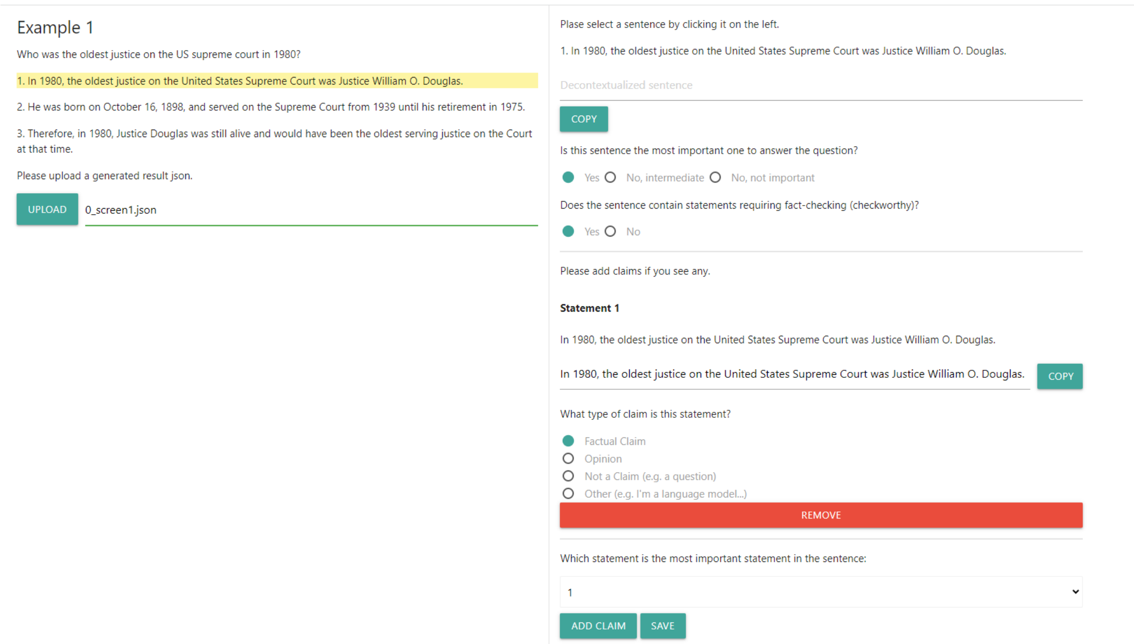


Figure 5-4: Screenshot of the first annotation interface for Decomposition, Decontextualization, and Check-worthiness detection.

Quality Control. To guarantee the annotation quality, instead of employing crowdsourcing annotators, we perform an in-house labelling by ten annotators who

[6]Without the annotation tool, on average, it takes ∼1.5 hours to annotate a 120-word response with about five bullet points, and more than 4 hours to annotate a 400-word response with ten bullet points, especially when the annotator is not familiar with details of an event (e.g., *What are some details that are public about the 2021 Capitol Hill riots*). With the tool, it takes 15-30 minutes to label a 50-150 words document.

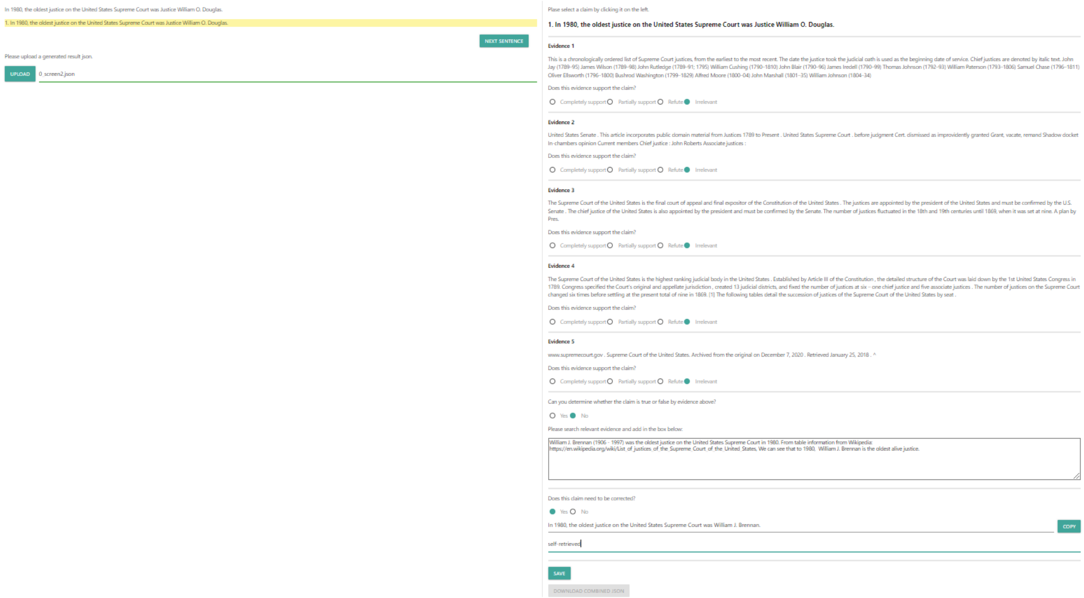


Figure 5-5: Screenshot of the second annotation interface: Evidence stance identification and Claim correction.

are Master's and PhD students, postdocs, and professors and are familiar with fact-checking.

Two annotators as a group are responsible for 22 model responses. For each step, annotators first independently finish individual annotations, and then consolidate their results with the group partner. In consolidation, partners discuss their disagreements until reaching a consensus. For cases where it is hard to reach an agreement even with the participation of the third rater, we discard it. Three steps are rigorously conducted serially. Annotators start the second step only after they finish the consolidation of the first step. Collecting evidence and judging stances is the most time- and patience-consuming step. To ensure quality, we incorporate the third rater when consolidating the second-step annotations in case of unintentional mistakes.

5.4 Annotation Guidelines

Based on the annotation framework, we aim to construct a benchmark for evaluation, assessing whether a fact-checking pipeline is effective in detecting and correcting factual errors in texts. The benchmark comprises annotations for the whole fact-

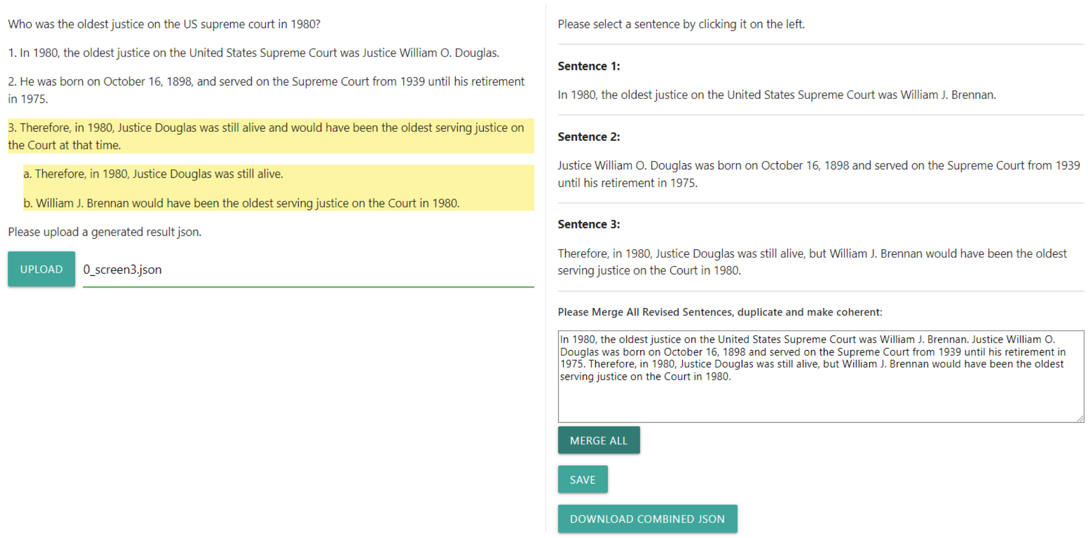


Figure 5-6: Screenshot of the third annotation interface: Claim Merge and Deduplication.

checking process spanning seven subtasks for 110 (question, response) pairs, in which most responses are generated by ChatGPT, and some are by GPT-4. This section introduces the annotation guidelines.

For each example, annotators are given a pair of (question, response). A response is either an answer generated by LLMs responding to users' question, or a document returned by LLMs according to users' request. Annotators are required to give outputs of each step shown in Figure 5-1. We describe how to annotate for component subtasks throughout the pipeline, particularly clarifying how to deal with possible ambiguous scenarios.

5.4.1 Decompose into Atomic Claims

It is subjective to decide the granularity of decomposition. We may aim to break down a long document into a set of atomic claims, while the definition of an atomic claim varies. Here, we practically apply the following strategy:

- Start by decomposing into single sentences.

- If the sentence contains too much information, break it into several components, but annotators do not overdo it, e.g., decomposing *Capitol Hill riots happened on January 6, 2021* to one claim for a year and one for the day.

- If several pieces of information are strongly dependent on each other, they are expected to co-occur in one snippet of evidence text, no more breaking-down is needed.

5.4.2 Decontextualise Atomic Claims

The criteria of decontextualisation are to ensure that all separated statements fully preserve semantics presented in the original context. For example, a statement that *it happened on Jan 6, 2021* loses information in decomposition, which makes it uncheckable. In such cases, annotators should replace pronouns, such as *it, they, those, these, this, that*, with specific entities or events after decomposition. Decontextulisation is mostly needed over cases with coreference relation. For complex relations, such as two sentences are strongly dependent on each other, we encourage going back to the step of decomposition and keep the original text without breaking-down.

5.4.3 Identify Checkworthy Claims

We consider two aspects in check-worthiness identification:

- If a statement presents subjective opinions, then it is not check-worthy.
- If the objective facts presented in a statement are commonsense, as obvious as *sun rises from the east*, it is not worth checking.

Therefore, we regard a statement as a check-worthy claim when it presents objective facts, and these facts are not apparent commonsense. There is a special case. If the objective facts presented in a statement are not publicly available information. Namely, we cannot collect any evidence over web pages related to the claim, such as personal experience. They are regarded as uncheckable claims.

Specifically, for each statement, annotators are asked to answer two questions. Which category does this claim fall into? (1) factual claim; (2) subjective opinion; (3) not a claim; and (4) other. Is this statement worth checking? (1) Yes; and (2) No.

5.4.4 Retrieve and Collect Evidence

Given a checkworthy claim, annotators are asked to search and collect the five most relevant snippets of text as evidence based on general web pages (including Wikipedia pages). Annotators are allowed to use any form of queries in retrieval, e.g. questions covering some aspects of the claim, or entities in the claim, and they need to record all queries and indicate those used for searching the most relevant evidence.

Note that five pieces of evidence is not a hard criterion. If less than five (even only one) pieces of evidence are sufficient to verify the input claim, and they are from reliable sources, annotators are allowed to collect $<$5 results. Meanwhile, if a claim involves a controversial topic, annotators are also encouraged to collect more than five results.

For each piece of evidence, record meta-data including (1) corresponding query, (2) citation (URL) of the web page from which this piece of evidence is extracted, (3) judgement of whether the source of evidence is reliable or not[7], and (4) indicator whether this individual evidence is sufficient to verify the input claim.

The aforementioned guidelines are applicable to claims for which there exists evidence over web pages. However, there are situations where there is not any information on public web pages, e.g. personal experience. They are objective facts, but are not extensively known by the public. Put differently, they are uncheckable. Annotators can give an empty list of evidence for uncheckable claims.

5.4.5 Identify Evidence Stance and Determine Correction

Given a claim and five pieces of most relevant evidence, annotators judge whether the evidence supports, partially supports, refutes or is irrelevant to the claim. For a claim, there will be K snippets of text (evidence), corresponding stance vectors $[\mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_K]$ and source reliability values $[r_1, r_2, \ldots, r_K]$. We skip all irrelevant evidence and follow the criteria below to determine whether edits are needed for a claim:

- If the claim is completely supported by evidence, no edit.

[7]Source reliability can also automatically be collected from MBFC/AllSides/Politifact/, but they apply for a small number of sources.

- If the claim is completely refuted by evidence, check the evidence and make edits accordingly one by one.

- If some evidence supports the claim and some refute it, this means there are conflicts between evidence (see Figure 5.1 for example). In such a scenario, we consider both the source reliability and the number of evidence falling into each stance. If the voice of "refute" is stronger than "support", we edit, otherwise remains the original text.

- If some refute and some partially support, there are two possible situations depending on whether the supported partition is the same as the refuted partition: (1) if what is supported and what is refuted are the same partition, there are conflicts between evidence, follow the steps above; and (2) if they support and refute different partition of the claim, edit the refuted partition.

5.4.6 Edit, Merge and Deduplicate

In correction, we keep the principle of making minimal edits against the original text to correct factual errors. Annotators do not add extra information provided by evidence that is not directly targeted at factual errors. No extra deletion, insertion or addition. Finally, annotators merge all statements, either revised or original ones, in order, and deduplicate repeated information with the principle of minimal edits.

5.5 Benchmark Analysis

During annotation, we remove another 16 examples because there is no standard gold answer for these questions, such as seven involving a flow of procedures, six non-factual questions, one tricky riddle-like question, one broken generated answer, and one highly-disagreed case, resulting in 94 instances. Statistics are shown in Table 5.2.

From the perspective of LLM users, we may expect to assess any answers and identify whether they are true and reliable, including the cases deleted in our setting. It should be highlighted that the questions involving a flow of procedures, tricky

	document	sent	cw_sent	claim	cw_claim	evid
size	94	311	277	678	661	3,305

Table 5.2: Statistics of the dataset. cw_sent = check-worthy sentences, cw_claim = check-worthy claims, evid = the total pieces of evidence, five for each cw_claim.

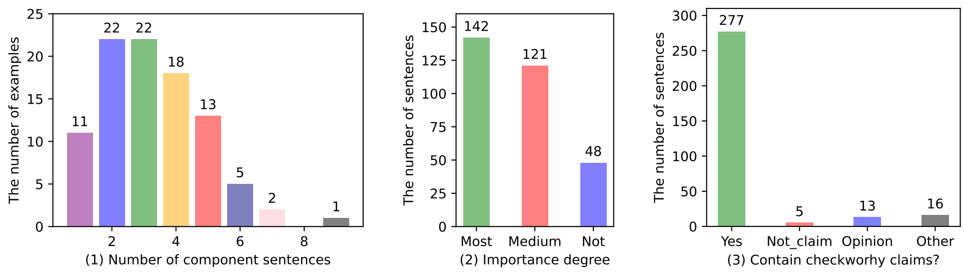


Figure 5-7: **Sentence analysis:** (1) Distribution of the number of sentences for each response; (2) Importance degree of sentences to answer the question (The distribution of the most important sentences to answer the question, intermediate important and not important; (3) The number of sentences across four types in terms of whether the sentence contains statements requiring fact-checking, Not_claim refers to *not a claim*, such as a question.

riddles, or non-factual questions need to be verified, while they are just out of the verification scope of the current fact-checking pipelines that only concern facts. The causality and the global logic behind the whole answer are under-explored.

Sentences: Most responses contain 2–5 sentences, with the longest response encompassing 9 sentences as shown in Figure 5-7. 142 sentences are considered to be the most important sentences, 121 and 48 fall into intermediate and not important. 278 sentences contain check-worthy statements, 16, 12 and 5 are categorized into *other*, *opinion*, and *not a claim*, respectively.

How does the evidence support the claim? Two-thirds of the evidence is irrelevant (2057/3305). We compare the stance distribution of claims in which automatically-retrieved evidence is enough to determine its factuality and the claims that cannot be determined by automatic evidence in Figure 5-9. Though the majority of evidence are irrelevant for both groups, there are only 17 strong-position stances ("completely support": *CS* and "refute": *RE*) in the latter, compared with 690 *CS* and 150 *RE* in the former.

Source	In-house	Closed-QA	Open-QA	All
Collected	45	33	35	115
Annotated	39	30	25	94
False	38	16	8	61

Table 5.3: False responses over three question sources.

Statistics. 277 sentences contain factual statements among 311. There are 678 atomic claims, where 661 claims are checkworthy, 16 are opinions and one is *not-a-claim*. For each check-worthy claim, five pieces of evidence are collected, resulting in 3,305 (claim, evidence, stance) triplets.

How many examples are factually correct? 61 examples contain factual errors, and 31 are factually correct, 2 without check-worthy claims. Amongst, 53 examples contain false claims, and 19 examples contain claims in which annotators cannot verify the statement due to insufficient evidence despite the manual search. Generally, one example contains 0-5 false claims. There are six examples with >5 incorrect claims. 16 sentences among 331 are deleted. 12 are total hallucinations, e.g., *Trump was the second black president.* 4 sentences are removed due to lacking enough evidence to support its factual correctness.

Table 5.3 shows that more incorrect responses appear in in-house collected questions, followed by Dolly closed questions that require knowledge to obtain a unique correct answer. Fewer errors occur in Dolly open questions, in which correct answers are not unique, e.g., *How do you play an E major chord on a guitar?* It has diverse correct answers requiring more general knowledge.

Claims. Of 678 claims, 419 and 227 are labelled as the most and intermediate important claims, and only 32 fall into *not-important*, indicating that users concern with almost the whole response given their importance. We analyze annotations of 661 check-worthy claims from two perspectives.

Can raters determine the factuality of a claim depending on the automatically-collected evidence? For 439 claims, annotators can determine *true or false* with automatic evidence, while 222 claims (one-third) need further manual retrieval to make judgements. Among the 222 claims, 125 true claims fall into

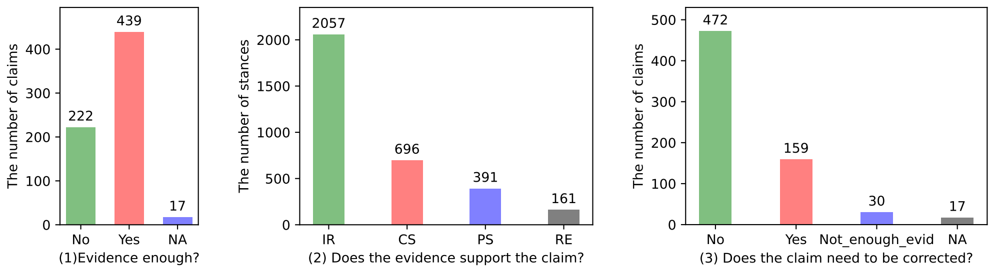


Figure 5-8: **Claim analysis:** (1) whether raters can determine the factuality of a claim depending on the automatically-collected evidence (*Yes/No*); (2) does the evidence support the claim (*CP*: completely support, *PS*: partially support, *RE*: refute, *IR*: irrelevant); (3) does the claim need to be corrected. NA (17) refers to 16 opinion-claims + 1 *not-a-claim*.

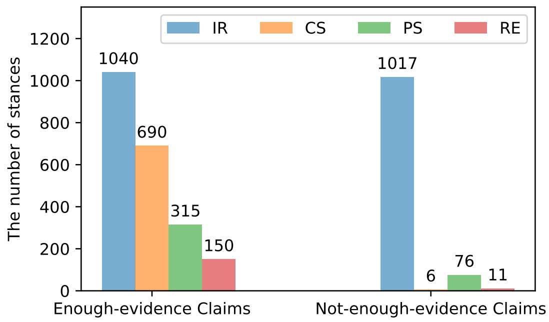


Figure 5-9: **Stance distribution** of claims with enough automatically-retrieved evidence to determine the factuality vs. claims without enough evidence (*CP*: completely support, *PS*: partially support, *RE*: refute, *IR*: irrelevant).

domain knowledge and information that is less known by the external people given a country, region, company, or an individual. The other half are either factually-incorrect claims (76) or undetermined claims without sufficient evidence despite manual retrieval (21), shown in Figure 5-9.

This suggests the ineffectiveness of the automatic evidence retrieval methods on collecting rare knowledge and evidence conditioned on false premises (claims). However, it may also reply that not all facts have been presented by textual descriptions directly. Some facts are unknown by the public, and some require connecting and reasoning knowledge from multiple sources, e.g., *did Aristotle use a laptop?* [Geva et al., 2021].

How many claims need to be corrected? In Figure 5-8, about a quarter (159/661) of claims are factually incorrect and need to be corrected. 30 claims are undetermined due to inadequate related information and knowledge even with manual retrieval. It is hard to obtain reliable related information about these cases by searching publicly-available sources. They involve expert-level knowledge (e.g., gene, water memory, black hole) and private details of an individual, organization, or country (personal awards and preferences, revenue of a company), which are only known by a small group of people, such as domain experts or internal individuals who are familiar with the event.

Original vs. revised responses. We quantify the difference between the original responses and the human-revised responses over the 61 false responses, showing that the normalized edit distance is 0.354, word overlap is 0.715, while semantically, BERTScore-F1 is 0.955 and cosine similarity based on SimCSE (Roberta-large) is 0.912. This implies that the core content of LLM answers is mostly correct, but minor factual mistakes are easily made by LLMs in detail, leading to high semantic similarity but multiple lexical edits in small errors.

Summary. The dataset consists of 94 ChatGPT (prompt, response) pairs. Each sample has detailed labels concerning the verification: elements of decontextualized sentences, atomic claims, the importance degree of the sentence, claim to the response, five pieces of evidence for a claim, the relationship between a claim and evidence, factual label (*true or false*) and revised version of claims, sentences, and the response.

Also, we compare the results of automatic methods that are commonly used in current fact-checking systems (e.g., RARR, FActScore, FacTool) for subtasks with human annotations. We first compare the automatic and human-annotated decomposition of atomic claim, and then evaluate five subtasks: (1) identify whether the sentence contains a factual statement; (2) detect the check-worthiness of a claim by categories of *factual*, *opinion*, *not a claim* and *other*; (3) judge the stance of a given evidence against a claim, whether it *supports*, *partially supports*, *refutes* or is *irrelevant* to the claim; (4) determine whether a claim is factually true or false, give a claim without "gold evidence", if false, revise it into a correct one; (5) edit a list of originally-true or revised claims into a new response, given the original response, to correct the factual errors while preserving the linguistic features and style of the

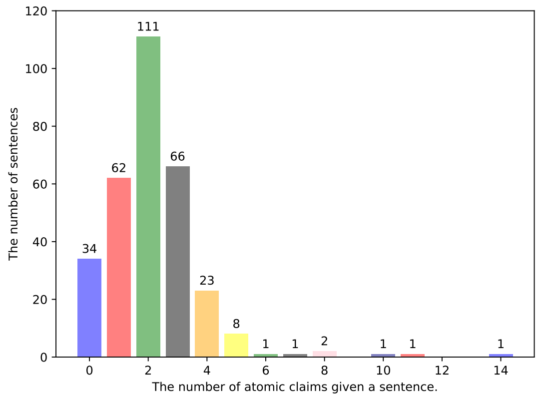


Figure 5-10: The distribution of component atomic claims amount given a sentence.

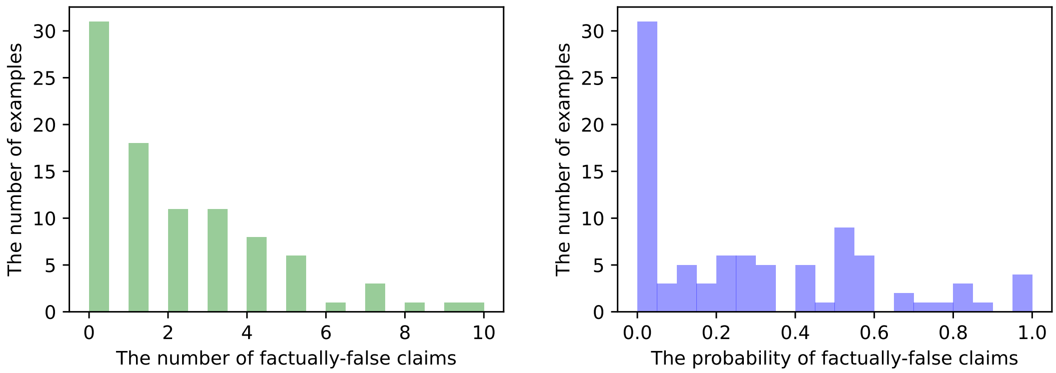


Figure 5-11: The number of false claims given an example.

original.

Key take-away is that ChatGPT shows promising results in atomic-claim decomposition, but low F1-score in check-worthiness detection. Also, verification remains challenging, especially when identifying false claims, even if it involves harnessing external knowledge. GPT-4 can generate sounding revised responses based on true statements. It's still an open-question in terms of how to evaluate the quality of revised responses by intrinsic metrics.

5.6 Conclusions

In this chapter we proposed a fine-grained annotation framework and constructed Factcheck-Bench[8], a benchmark to evaluate automatic fact-checkers of LLM output.

[8] https://github.com/yuxiaw/Document-level-fact-checking

The benchmark contains 678 open-domain claims generated by LLMs, involving annotations of eight subtasks for detecting and correcting the factual errors in long documents. Human annotations show that LLMs are prone to make factual errors in expert-level knowledge and exclusive details known by a small group of people. Fact-checking results show that current verifiers are struggling to identify open-domain false claims. We plan to explore this in future work.

Chapter 6

Conclusion

In this thesis, we have studied efficient uncertainty estimation methods for applied natural language processing tasks. Existing uncertainty estimation methods have their drawbacks related to the computation of uncertainty that is not relevant to the task, long computation time, poor generalizability to different types of data and models, lack of interpretability, and difficulty of use in practice. So, we have presented effective methods for uncertainty estimation and their application to two different NLP tasks, namely active learning and fact-checking.

In **Chapter 3** we proposed a new batch active learning algorithm **Large Batch-BALD (LBB)** that performs an approximation of the BatchBALD method, using the BALD acquisition function and pairwise mutual information of model output components. The proposed algorithm is as efficient in avoiding taking similar objects in one batch as the original method, while it computes the acquisition function several times faster, especially in the case of large batches. Thus, this active learning algorithm balances the uncertainty and diversity of the acquired samples and significantly reduces the acquisition time compared to the original BatchBALD. The resulting method is shown to be efficient for batch AL in application to modern text and image datasets. In addition, we implemented and benchmarked our algorithms LBB and PLBB (its stochastic extension) as well as its main competitor PBALD within the **ALToolbox** framework, thus making the methods more user-friendly and emphasizing the industrial appeal of its use.

In **Chapter 4** we presented a novel approach to fact-checking and hallucination detection based on token-level uncertainty quantification. According to human evaluation, our approach is competitive with FactScore, a fact-checking tool that leverages an external knowledge source: we achieve similar or better results with access to only LLM outputs. Then, we proposed a computationally efficient token-level and claim-level UQ method, **Claim Conditioned Probability**, that outperforms a number of baselines in fact-checking. In this method, we post-process the word distribution to mitigate the impact of uncertainty related to the variability of surface forms and uncertainty about what claim type to generate on the current step. In the constructed benchmark, where we detect hallucinations in biographies, CCP outperforms other methods for seven LLMs, including GPT-3.5-turbo and GPT-4, and four languages. We also demonstrate that computational overhead of CCP might be as low as 3% of the LLM inference runtime.

Finally, in **Chapter 5** we proposed **Factcheck-Bench**, a holistic end-to-end framework for annotating and evaluating the factuality of LLM-generated responses. It encompasses a multi-stage annotation scheme designed to yield detailed labels for fact-checking and correcting not just the final prediction, but also the intermediate steps that a fact-checking system might need to take. Based on this framework, we construct an open-domain factuality benchmark in three-levels of granularity: claim, sentence, and document. It covers both detection and revision, to facilitate the evaluation and the analysis of automatic fact-checking systems and their internal components.

To sum up the section and this thesis in general, we provide answers to the research questions we stated in Section 1.1:

Q1: Can we design a method for quantifying uncertainty for batch active learning that can effectively choose multiple examples to label at each step? Yes, we can design such a method based on diversity provided by design of the algorithm. It is possible with approximation of BatchBALD method that incorporates both uncertainty and diversity inside. It makes a new method comparable in quality and much faster in inference than the original algorithm.

Q2: Can we develop an uncertainty estimation method that distinguishes between different types of uncertainty in a generative language model? Yes, using the proposed Claim Conditioned Probability method, we can detect and exclude irrelevant uncertainty and base our measure only on the uncertainty of the target claim. It makes the process of fact-checking more efficient and precise.

Future directions

Based on the obtained results, we see several directions for future research. Regarding active learning, we see that acquisition functions lack versatility, namely high performance on different datasets and models. Often, basic methods with simple intuition perform better than complex algorithms, especially on complex data and large acquisition batches. Regarding fact-checking, we see that existing methods do not fully account for all the hallucination scenarios created by LLMs. Thus, the quality of false content detection is reduced on complex or previously unseen examples. We propose to move towards generalizing the behavior of uncertainty estimation methods to achieve universality in their performance. Adding theoretical guarantees for the methods used, as well as extensive experimental comparisons, can help.

Also, we see significant potential in using uncertainty measures for decision-making as a future research direction. Decisions based on the values of uncertainty measures are often ambiguous. The absolute values obtained can highlight places that should be investigated in more detail, but some threshold is needed to make decisions based on these values. Such an approach can be addressed by calibration, for example, as is done in conformal prediction procedure. The threshold is calculated based on the hold-out calibration set and then applied to test samples to decide which target to include in the so-called prediction set, which has formal guarantees regarding the final prediction quality. Thus, the combination of large language models with calibration procedures, and in particular with conformal prediction, is of considerable interest.

Limitations

Scalable Batch Acquisition for Deep Bayesian Active Learning. The method is applied to pool-based active learning for a classification task. As in classical active learning, the model is trained from scratch at each step. Issues related to additional pre-training of the model or fine-tuning of the pre-trained model using active learning steps are beyond the scope of this setting.

Fact-Checking the Output of Large Language Models via Token-Level Uncertainty Quantification. While this work has been conducted according to experimental and methodological best practice, there are several potential limitations. First, the current implementation of the method makes use of OpenAI's GPT models for text segmentation and extraction of atomic facts, similarly to FactScore, which may not be practical in real applications. Replacing these components with cheaper open models should be feasible in principle, but is left for future work. Second, part of our experimental results rely on a human evaluation, which may be subjective. We tried to mitigate this by creating detailed instructions for the annotators, but a larger-scale study with larger overlap would further strengthen the results. Third, our uncertainty quantification is based on tokens, where taking into account also larger units such as noun or verb phrases as basic units of analysis may be better motivated linguistically. Finally, our approach only detects potentially spurious generations. A prominent direction for future research is to modify the generation of an LLM to exclude such spans, while ensuring fluency of a generated text: a simple removal of uncertain claims may result in incoherent sentences.

Factcheck-Bench: Fine-Grained Evaluation Benchmark for Automatic Fact-Checkers The current version of dataset consists of only 94 (question, response) pairs, we plan to scale up the dataset in English, Chinese, and Arabic in future work. More than half of automatically retrieved evidences are irrelevant. Improving the relevance of retrieved evidence is critical to the accuracy of fact-checking. The automatic atomic-claim decomposition and evidence retrieval systems incorporated in the fact-checking annotations may introduce biases, which can affect the annotation results. Besides, the dataset does not cover all types of claims. Limited scope and coverage may be more effective in certain domains, possibly leading to inaccurate or unfair assessments in certain domains for automatic fact-checkers. The responses generated by LLMs might also inherit some biases present in the involved LLMs.

Bibliography

Anastasios N Angelopoulos and Stephen Bates. A gentle introduction to conformal prediction and distribution-free uncertainty quantification. *arXiv preprint arXiv:2107.07511*, 2021. URL https://arxiv.org/abs/2107.07511.

Jordan Ash, Surbhi Goel, Akshay Krishnamurthy, and Sham Kakade. Gone fishing: Neural active learning with fisher embeddings. In *Advances in Neural Information Processing Systems*, volume 34, pages 8927–8939. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper/2021/file/4afe044911ed2c247005912512ace23b-Paper.pdf.

Jordan T. Ash, Chicheng Zhang, Akshay Krishnamurthy, John Langford, and Alekh Agarwal. Deep batch active learning by diverse, uncertain gradient lower bounds. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=ryghZJBKPS.

Pepa Atanasova, Jakob Grue Simonsen, Christina Lioma, and Isabelle Augenstein. Generating fact checking explanations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7352–7364, Online, July 2020. Association for Computational Linguistics. doi:10.18653/v1/2020.acl-main.656. URL https://aclanthology.org/2020.acl-main.656.

Pepa Atanasova, Jakob Grue Simonsen, Christina Lioma, and Isabelle Augenstein. Fact Checking with Insufficient Evidence. *Transactions of the Association for Computational Linguistics*, 10:746–763, 2022. doi:10.1162/tacl_a_00486. URL https://aclanthology.org/2022.tacl-1.43.

Parmida Atighehchian, Frédéric Branchaud-Charron, and Alexandre Lacoste. Bayesian active learning for production, a systematic study and a reusable library. *arXiv preprint arXiv:2006.09916*, 2020.

Isabelle Augenstein. *Towards Explainable Fact Checking*. Dr. Scient. thesis, University of Copenhagen, Faculty of Science, 2021. URL https://arxiv.org/abs/2108.10274.

Isabelle Augenstein, Christina Lioma, Dongsheng Wang, Lucas Chaves Lima, Casper Hansen, Christian Hansen, and Jakob Grue Simonsen. MultiFC: A real-world multi-domain dataset for evidence-based fact checking of claims. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4685–4697, Hong Kong, China, November 2019.

Association for Computational Linguistics. doi:10.18653/v1/D19-1475. URL https://aclanthology.org/D19-1475.

Yejin Bang, Samuel Cahyawijaya, Nayeon Lee, Wenliang Dai, Dan Su, Bryan Wilie, Holy Lovenia, Ziwei Ji, Tiezheng Yu, Willy Chung, Quyet V. Do, Yan Xu, and Pascale Fung. A multitask, multilingual, multimodal evaluation of ChatGPT on reasoning, hallucination, and interactivity. In *Proceedings of the 13th International Joint Conference on Natural Language Processing and the 3rd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 675–718, Nusa Dua, Bali, November 2023. Association for Computational Linguistics. URL https://aclanthology.org/2023.ijcnlp-main.45.

Oscar Barrera, Sergei Guriev, Emeric Henry, and Ekaterina Zhuravskaya. Facts, alternative facts, and fact checking in times of post-truth politics. *Journal of public economics*, 182:104123, 2020. URL https://www.sciencedirect.com/science/article/pii/S0047272719301859.

William H Beluch, Tim Genewein, Andreas Nürnberger, and Jan M Köhler. The power of ensembles for active learning in image classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 9368–9377, 2018. URL https://openaccess.thecvf.com/content_cvpr_2018/papers/Beluch_The_Power_of_CVPR_2018_paper.pdf.

Steven Bird and Edward Loper. NLTK: The natural language toolkit. In *Proceedings of the ACL Interactive Poster and Demonstration Sessions*, pages 214–217, Barcelona, Spain, July 2004. Association for Computational Linguistics. URL https://aclanthology.org/P04-3031.

Samuel Budd, Emma C Robinson, and Bernhard Kainz. A survey on active learning and human-in-the-loop deep learning for medical image analysis. *Medical image analysis*, 71:102062, 2021. URL https://www.sciencedirect.com/science/article/abs/pii/S1361841521001080.

Jannis Bulian, Jordan Boyd-Graber, Markus Leippold, Massimiliano Ciaramita, and Thomas Diggelmann. Climate-fever: A dataset for verification of real-world climate claims. In *NeurIPS 2020 Workshop on Tackling Climate Change with Machine Learning*, 2020. URL https://arxiv.org/abs/2012.00614.

Evgeny Burnaev and Maxim Panov. Adaptive design of experiments based on gaussian processes. In *Statistical Learning and Data Sciences*, pages 116–125, Cham, 2015. Springer International Publishing. URL https://link.springer.com/chapter/10.1007/978-3-319-17091-6_7.

Jifan Chen, Aniruddh Sriram, Eunsol Choi, and Greg Durrett. Generating literal and implied subquestions to fact-check complex claims. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 3495–3516, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics. doi:10.18653/v1/2022.emnlp-main.229. URL https://aclanthology.org/2022.emnlp-main.229.

Shiqi Chen, Yiran Zhao, Jinghan Zhang, I-Chun Chern, Siyang Gao, Pengfei Liu, and Junxian He. Felm: Benchmarking factuality evaluation of large language models. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2023. URL http://arxiv.org/abs/2310.00741.

I-Chun Chern, Steffi Chern, Shiqi Chen, Weizhe Yuan, Kehua Feng, Chunting Zhou, Junxian He, Graham Neubig, Pengfei Liu, et al. FacTool: Factuality detection in generative AI–a tool augmented framework for multi-task and multi-domain scenarios. *arXiv preprint arXiv:2307.13528*, 2023. URL https://arxiv.org/abs/2307.13528.

Eunsol Choi, Jennimaria Palomaki, Matthew Lamm, Tom Kwiatkowski, Dipanjan Das, and Michael Collins. Decontextualization: Making sentences stand-alone. *Transactions of the Association for Computational Linguistics*, 9:447–461, 2021. doi:10.1162/tacl_a_00377. URL https://aclanthology.org/2021.tacl-1.27.

Gui Citovsky, Giulia DeSalvo, Claudio Gentile, Lazaros Karydas, Anand Rajagopalan, Afshin Rostamizadeh, and Sanjiv Kumar. Batch active learning at scale. In *Advances in Neural Information Processing Systems*, volume 34, pages 11933–11944. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/64254db8396e404d9223914a0bd355d2-Paper.pdf.

Zhuyun Dai, Vincent Y Zhao, Ji Ma, Yi Luan, Jianmo Ni, Jing Lu, Anton Bakalov, Kelvin Guu, Keith Hall, and Ming-Wei Chang. Promptagator: Few-shot dense retrieval from 8 examples. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=gmL46YMpu2J.

David Dale, Elena Voita, Loic Barrault, and Marta R. Costa-jussà. Detecting and mitigating hallucinations in machine translation: Model internal workings alone do well, sentence similarity Even better. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 36–50, Toronto, Canada, July 2023. Association for Computational Linguistics. doi:10.18653/v1/2023.acl-long.3. URL https://aclanthology.org/2023.acl-long.3.

Tivadar Danka and Peter Horvath. modal: A modular active learning framework for Python. *arXiv preprint arXiv:1805.00979*, 2018.

Shehzaad Dhuliawala, Mojtaba Komeili, Jing Xu, Roberta Raileanu, Xian Li, Asli Celikyilmaz, and Jason E Weston. Chain-of-verification reduces hallucination in large language models. In *ICLR 2024 Workshop on Reliable and Responsible Foundation Models*, 2024. URL https://openreview.net/forum?id=ek3GqAs2uO.

Ekaterina Fadeeva, Roman Vashurin, Akim Tsvigun, Artem Vazhentsev, Sergey Petrakov, Kirill Fedyanin, Daniil Vasilev, Elizaveta Goncharova, Alexander Panchenko, Maxim Panov, Timothy Baldwin, and Artem Shelmanov. LM-Polygraph: Uncertainty estimation for language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 446–461, Singapore, December 2023. Association for Computational Linguistics. doi:10.18653/v1/2023.emnlp-demo.41. URL https://aclanthology.org/2023.emnlp-demo.41.

Ekaterina Fadeeva, **Aleksandr Rubashevskii**, Artem Shelmanov, Sergey Petrakov, Haonan Li, Hamdy Mubarak, Evgenii Tsymbalov, Gleb Kuzmin, Alexander Panchenko, Timothy Baldwin, Preslav Nakov, and Maxim Panov. Fact-checking the output of large language models via token-level uncertainty quantification. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 9367–9385, Bangkok, Thailand and virtual meeting, August 2024. Association for Computational Linguistics. doi:10.18653/v1/2024.findings-acl.558. URL https://aclanthology.org/2024.findings-acl.558.

Angela Fan, Aleksandra Piktus, Fabio Petroni, Guillaume Wenzek, Marzieh Saeidi, Andreas Vlachos, Antoine Bordes, and Sebastian Riedel. Generating fact checking briefs. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7147–7161, Online, November 2020. Association for Computational Linguistics. doi:10.18653/v1/2020.emnlp-main.580. URL https://aclanthology.org/2020.emnlp-main.580.

Marina Fomicheva, Shuo Sun, Lisa Yankovskaya, Frédéric Blain, Francisco Guzmán, Mark Fishel, Nikolaos Aletras, Vishrav Chaudhary, and Lucia Specia. Unsupervised quality estimation for neural machine translation. *Transactions of the Association for Computational Linguistics*, 8:539–555, 2020. doi:10.1162/tacl_a_00330. URL https://aclanthology.org/2020.tacl-1.35.

Yarin Gal. *Uncertainty in deep learning.* PhD thesis, University of Cambridge, 2016. URL https://www.semanticscholar.org/paper/Uncertainty-in-Deep-Learning-Gal/3c623c08329e129e784a5d03f7606ec8feba3a28.

Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 1050–1059, New York, New York, USA, 20–22 Jun 2016. PMLR. URL https://proceedings.mlr.press/v48/gal16.html.

Yarin Gal, Riashat Islam, and Zoubin Ghahramani. Deep Bayesian active learning with image data. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1183–1192. PMLR, 06–11 Aug 2017. URL https://proceedings.mlr.press/v70/gal17a.html.

Luyu Gao, Zhuyun Dai, Panupong Pasupat, Anthony Chen, Arun Tejasvi Chaganty, Yicheng Fan, Vincent Zhao, Ni Lao, Hongrae Lee, Da-Cheng Juan, and Kelvin Guu. RARR: Researching and revising what language models say, using language models. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki, editors, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 16477–16508, Toronto, Canada, July 2023. Association for Computational Linguistics. doi:10.18653/v1/2023.acl-long.910. URL https://aclanthology.org/2023.acl-long.910.

Mor Geva, Daniel Khashabi, Elad Segal, Tushar Khot, Dan Roth, and Jonathan Berant. Did aristotle use a laptop? A question answering benchmark with implicit reasoning strategies. *Transactions of the Association for Computational Linguistics*, 9:346–361, 2021. URL https://arxiv.org/abs/2101.02235.

Zhijiang Guo, Michael Schlichtkrull, and Andreas Vlachos. A survey on automated fact-checking. *Transactions of the Association for Computational Linguistics*, 10: 178–206, 2022. doi:10.1162/tacl_a_00454. URL https://aclanthology.org/2022.tacl-1.11.

Naeemul Hassan, Chengkai Li, and Mark Tremayne. Detecting check-worthy factual claims in presidential debates. In *Proceedings of the 24th ACM International Conference on Information and Knowledge Management, CIKM 2015, Melbourne, VIC, Australia, October 19 - 23, 2015*, pages 1835–1838. ACM, 2015. doi:10.1145/2806416.2806652. URL https://doi.org/10.1145/2806416.2806652.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. URL https://www.cv-foundation.org/openaccess/content_cvpr_2016/papers/He_Deep_Residual_Learning_CVPR_2016_paper.pdf.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. DeBERTa: decoding-enhanced BERT with disentangled attention. In *9th International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=XPZIaotutsD.

Steven C. H. Hoi, Rong Jin, Jianke Zhu, and Michael R. Lyu. Batch mode active learning and its application to medical image classification. In *Proceedings of the 23rd International Conference on Machine Learning*, ICML '06, page 417–424, New York, NY, USA, 2006. Association for Computing Machinery. ISBN 1595933832. doi:10.1145/1143844.1143897. URL https://doi.org/10.1145/1143844.1143897.

N. Houlsby, F. Huszár, et al. Bayesian active learning for classification and preference learning. *CoRR*, 1112.5745, 2011. URL https://arxiv.org/abs/1112.5745.

Robert Iv, Alexandre Passos, Sameer Singh, and Ming-Wei Chang. FRUIT: Faithfully reflecting updated information in text. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3670–3686, Seattle, United States, July 2022. Association for Computational Linguistics. doi:10.18653/v1/2022.naacl-main.269. URL https://aclanthology.org/2022.naacl-main.269.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023. URL https://arxiv.org/abs/2310.06825.

Maël Jullien, Marco Valentino, Hannah Frost, Paul O'regan, Donal Landers, and André Freitas. SemEval-2023 task 7: Multi-evidence natural language inference for clinical trial data. In *Proceedings of the 17th International Workshop on Semantic Evaluation (SemEval-2023)*, pages 2216–2226, Toronto, Canada, July 2023. Association for Computational Linguistics. doi:10.18653/v1/2023.semeval-1.307. URL https://aclanthology.org/2023.semeval-1.307.

Saurav Kadavath, Tom Conerly, Amanda Askell, Tom Henighan, Dawn Drain, Ethan Perez, Nicholas Schiefer, et al. Language models (mostly) know what they know. *arXiv preprint arXiv:2207.05221*, 2022. doi:10.48550/arXiv.2207.05221. URL https://doi.org/10.48550/arXiv.2207.05221.

Adam Tauman Kalai and Santosh S. Vempala. Calibrated language models must hallucinate. In *Proceedings of the 56th Annual ACM Symposium on Theory of Computing*, page 160–171, New York, NY, USA, 2024. Association for Computing Machinery. doi:10.1145/3618260.3649777. URL https://doi.org/10.1145/3618260.3649777.

Ryo Kamoi, Tanya Goyal, Juan Rodriguez, and Greg Durrett. WiCE: Real-world entailment for claims in Wikipedia. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 7561–7583, 2023. URL https://aclanthology.org/2023.emnlp-main.470.

Andreas Kirsch, Joost Van Amersfoort, and Yarin Gal. Batchbald: Efficient and diverse batch acquisition for deep bayesian active learning. *Advances in neural information processing systems*, 32, 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/95323660ed2124450caaac2c46b5ed90-Paper.pdf.

Andreas Kirsch, Sebastian Farquhar, Parmida Atighehchian, Andrew Jesson, Frédéric Branchaud-Charron, and Yarin Gal. Stochastic batch acquisition: A simple baseline for deep active learning. *Transactions on Machine Learning Research*, 2021. URL https://openreview.net/forum?id=vcHwQyNBjW.

Nikita Kotelevskii, Aleksandr Artemenkov, Kirill Fedyanin, Fedor Noskov, Alexander Fishkov, Artem Shelmanov, Artem Vazhentsev, Aleksandr Petiushko, and Maxim Panov. Nonparametric uncertainty quantification for single deterministic neural network. *Advances in Neural Information Processing Systems*, 35:36308–36323, 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/eb7389b039655fc5c53b11d4a6fa11bc-Abstract-Conference.html.

A. Krizhevsky. Learning multiple layers of features from tiny images. *Technical report*, 2009. URL https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf.

Lorenz Kuhn, Yarin Gal, and Sebastian Farquhar. Semantic uncertainty: Linguistic invariances for uncertainty estimation in natural language generation. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023. URL https://openreview.net/pdf?id=VD-AYtP0dve.

Philippe Laban, Tobias Schnabel, Paul N. Bennett, and Marti A. Hearst. SummaC: Re-Visiting NLI-based Models for Inconsistency Detection in Summarization. *Transactions of the Association for Computational Linguistics*, 10:163–177, 02 2022. ISSN 2307-387X. doi:10.1162/tacl_a_00453. URL https://doi.org/10.1162/tacl_a_00453.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NeurIPS 2017, page 6405–6416, Red Hook, NY, USA, 2017. Curran Associates Inc. ISBN 9781510860964. URL https://proceedings.neurips.cc/paper/2017/hash/9ef2ed4b7fd2c810847ffa5fa85bce38-Abstract.html.

Moritz Laurer, Wouter Van Atteveldt, Andreu Casas, and Kasper Welbers. Less Annotating, More Classifying: Addressing the Data Scarcity Issue of Supervised Machine Learning with Deep Transfer Learning and BERT-NLI. *Political Analysis*, pages 1–33, June 2023. ISSN 1047-1987, 1476-4989. doi:10.1017/pan.2023.20. URL https://www.cambridge.org/core/product/identifier/S1047198723000207/type/journal_article.

Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11), 1998. URL https://ieeexplore.ieee.org/document/726791.

Kimin Lee, Kibok Lee, Honglak Lee, and Jinwoo Shin. A simple unified framework for detecting out-of-distribution samples and adversarial attacks. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper_files/paper/2018/file/abdeb6f575ac5c6676b747bca8d09cc2-Paper.pdf.

David D Lewis and William A Gale. A sequential algorithm for training text classifiers. In *SIGIR'94*, pages 3–12. Springer, 1994. URL https://link.springer.com/chapter/10.1007/978-1-4471-2099-5_1.

Junyi Li, Xiaoxue Cheng, Xin Zhao, Jian-Yun Nie, and Ji-Rong Wen. HaluEval: A large-scale hallucination evaluation benchmark for large language models. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 6449–6464, Singapore, December 2023. Association for Computational Linguistics.

doi:10.18653/v1/2023.emnlp-main.397. URL https://aclanthology.org/2023.emnlp-main.397.

Bill Yuchen Lin, Dong-Ho Lee, Frank F. Xu, Ouyu Lan, and Xiang Ren. AlpacaTag: An active learning-based crowd annotation framework for sequence tagging. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 58–63, Florence, Italy, July 2019. Association for Computational Linguistics. doi:10.18653/v1/P19-3010. URL https://aclanthology.org/P19-3010.

Zhen Lin, Shubhendu Trivedi, and Jimeng Sun. Generating with confidence: Uncertainty quantification for black-box large language models. *Transactions on Machine Learning Research*, 2024. URL https://openreview.net/forum?id=DWkJCSxKU5.

Tianyu Liu, Xin Zheng, Baobao Chang, and Zhifang Sui. Towards faithfulness in open domain table-to-text generation from an entity-centric view. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 13415–13423. AAAI Press, 2021. URL https://ojs.aaai.org/index.php/AAAI/article/view/17583.

Yixin Liu, Alex Fabbri, Pengfei Liu, Yilun Zhao, Linyong Nan, Ruilin Han, Simeng Han, Shafiq Joty, Chien-Sheng Wu, Caiming Xiong, and Dragomir Radev. Revisiting the gold standard: Grounding summarization evaluation with robust human evaluation. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4140–4170, Toronto, Canada, July 2023. Association for Computational Linguistics. doi:10.18653/v1/2023.acl-long.228. URL https://aclanthology.org/2023.acl-long.228.

Tong Luo, K. Kramer, S. Samson, A. Remsen, D.B. Goldgof, L.O. Hall, and T. Hopkins. Active learning to recognize multiple types of plankton. In *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.*, volume 3, pages 478–481 Vol.3, 2004. doi:10.1109/ICPR.2004.1334570.

Xiaoming Lv, Fa jie Duan, Jia jia Jiang, Xiao Fu, and Lin Gan. Deep active learning for surface defect detection. *Sensors (Basel, Switzerland)*, 20, 2020. URL https://api.semanticscholar.org/CorpusID:213186633.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA, June 2011. Association for Computational Linguistics. URL https://aclanthology.org/P11-1015.

Andrey Malinin and Mark Gales. Uncertainty estimation in autoregressive structured prediction. In *International Conference on Learning Representations*, 2020. URL https://arxiv.org/abs/2002.07650.

Potsawee Manakul, Adian Liusie, and Mark Gales. SelfCheckGPT: Zero-resource black-box hallucination detection for generative large language models. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 9004–9017, Singapore, December 2023. Association for Computational Linguistics. doi:10.18653/v1/2023.emnlp-main.557. URL https://aclanthology.org/2023.emnlp-main.557.

Katerina Margatina, Giorgos Vernikos, Loïc Barrault, and Nikolaos Aletras. Active learning by acquiring contrastive examples. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 650–663, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi:10.18653/v1/2021.emnlp-main.51. URL https://aclanthology.org/2021.emnlp-main.51.

Alejandro Martín, Javier Huertas-Tato, Álvaro Huertas-García, Guillermo Villar-Rodríguez, and David Camacho. Facter-check: Semi-automated fact-checking through semantic similarity and natural language inference. *Knowledge-Based Systems*, 251:109265, 2022. ISSN 0950-7051. doi:https://doi.org/10.1016/j.knosys.2022.109265. URL https://www.sciencedirect.com/science/article/pii/S0950705122006323.

Joshua Maynez, Shashi Narayan, Bernd Bohnet, and Ryan McDonald. On faithfulness and factuality in abstractive summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1906–1919, Online, July 2020. Association for Computational Linguistics. doi:10.18653/v1/2020.acl-main.173. URL https://aclanthology.org/2020.acl-main.173.

Sewon Min, Kalpesh Krishna, Xinxi Lyu, Mike Lewis, Wen-tau Yih, Pang Koh, Mohit Iyyer, Luke Zettlemoyer, and Hannaneh Hajishirzi. FActScore: Fine-grained atomic evaluation of factual precision in long form text generation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 12076–12100, 2023. doi:10.18653/v1/2023.emnlp-main.741. URL https://aclanthology.org/2023.emnlp-main.741.

Dor Muhlgay, Ori Ram, Inbal Magar, Yoav Levine, Nir Ratner, Yonatan Belinkov, Omri Abend, Kevin Leyton-Brown, Amnon Shashua, and Yoav Shoham. Generating benchmarks for factuality evaluation of language models. *CoRR*, abs/2307.06908, 2023. doi:10.48550/arXiv.2307.06908. URL https://doi.org/10.48550/arXiv.2307.06908.

Niels Mündler, Jingxuan He, Slobodan Jenko, and Martin Vechev. Self-contradictory hallucinations of large language models: Evaluation, detection and mitigation. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=EmQSOi1X2f.

Preslav Nakov, David P. A. Corney, Maram Hasanain, Firoj Alam, Tamer Elsayed, Alberto Barrón-Cedeño, Paolo Papotti, Shaden Shaar, and Giovanni Da San Martino. Automated fact-checking for assisting human fact-checkers. In *Proceedings* *of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI 2021, Virtual Event / Montreal, Canada, 19-27 August 2021*, pages 4551–4558. ijcai.org, 2021a. doi:10.24963/ijcai.2021/619. URL https://doi.org/10.24963/ijcai.2021/619.

Preslav Nakov, Giovanni Da San Martino, Tamer Elsayed, Alberto Barrón-Cedeno, Rubén Míguez, Shaden Shaar, Firoj Alam, Fatima Haouari, Maram Hasanain, Nikolay Babulkov, et al. The clef-2021 checkthat! lab on detecting check-worthy claims, previously fact-checked claims, and fake news. In *Advances in Information Retrieval: 43rd European Conference on IR Research, ECIR 2021, Virtual Event, March 28–April 1, 2021, Proceedings, Part II 43*, pages 639–649. Springer, 2021b. URL https://arxiv.org/abs/2109.12987.

Yuval Netzer, Tao Wang, Adam Coates, A. Bissacco, Bo Wu, and A. Ng. Reading digits in natural images with unsupervised feature learning. In *Neural Information Processing Systems Workshop on Deep Learning and Unsupervised Feature Learning*, 2011. URL https://api.semanticscholar.org/CorpusID:16852518.

Minh-Quoc Nghiem, Paul Baylis, and Sophia Ananiadou. Paladin: an annotation tool based on active and proactive learning. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pages 238–243, Online, April 2021. Association for Computational Linguistics. doi:10.18653/v1/2021.eacl-demos.28. URL https://aclanthology.org/2021.eacl-demos.28.

Yixin Nie, Haonan Chen, and Mohit Bansal. Combining fact extraction and verification with neural semantic matching networks. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 6859–6866. AAAI Press, 2019. doi:10.1609/aaai.v33i01.33016859. URL https://doi.org/10.1609/aaai.v33i01.33016859.

Aleksandr Nikolich, Konstantin Korolev, and Artem Shelmanov. Vikhr: The family of open-source instruction-tuned large language models for russian. *arXiv preprint arXiv:2405.13929*, 2024. URL https://arxiv.org/abs/2405.13929.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744, 2022. URL https://arxiv.org/abs/2203.02155.

Liangming Pan, Xiaobao Wu, Xinyuan Lu, Anh Tuan Luu, William Yang Wang, Min-Yen Kan, and Preslav Nakov. Fact-checking complex claims with program-guided reasoning. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6981–7004,

2023. doi:10.18653/v1/2023.acl-long.386. URL https://aclanthology.org/2023.acl-long.386.

Amandalynne Paullada, Inioluwa Deborah Raji, Emily M. Bender, Emily Denton, and Alex Hanna. Data and its (dis)contents: A survey of dataset development and use in ML research. *Patterns*, 2(11), 2021. ISSN 2666-3899. doi:https://doi.org/10.1016/j.patter.2021.100336. URL https://www.sciencedirect.com/science/article/pii/S2666389921001847.

Vincent Plassier, Mehdi Makni, **Aleksandr Rubashevskii**, Eric Moulines, and Maxim Panov. Conformal prediction for federated uncertainty quantification under label shift. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 27907–27947. PMLR, 23–29 July 2023. URL https://proceedings.mlr.press/v202/plassier23a.html.

Vincent Plassier, Nikita Kotelevskii, **Aleksandr Rubashevskii**, Fedor Noskov, Maksim Velikanov, Alexander Fishkov, Samuel Horvath, Martin Takac, Eric Moulines, and Maxim Panov. Efficient conformal prediction under data heterogeneity. In *Proceedings of The 27th International Conference on Artificial Intelligence and Statistics*, volume 238 of *Proceedings of Machine Learning Research*, pages 4879–4887. PMLR, 02–04 May 2024. URL https://proceedings.mlr.press/v238/plassier24a.html.

Nicolas Pröllochs. Community-based fact-checking on twitter's birdwatch platform. In *Proceedings of the Sixteenth International AAAI Conference on Web and Social Media, ICWSM 2022, Atlanta, Georgia, USA, June 6-9, 2022*, pages 794–805. AAAI Press, 2022. URL https://ojs.aaai.org/index.php/ICWSM/article/view/19335.

Carl Edward Rasmussen. *Gaussian processes for machine learning*. MIT Press, 2006. URL https://gaussianprocess.org/gpml/.

Vikas Raunak, Arul Menezes, and Marcin Junczys-Dowmunt. The curious case of hallucinations in neural machine translation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1172–1183, Online, June 2021. Association for Computational Linguistics. doi:10.18653/v1/2021.naacl-main.92. URL https://aclanthology.org/2021.naacl-main.92.

Nils Reimers and Iryna Gurevych. Sentence-BERT: Sentence embeddings using siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 11 2019. URL https://arxiv.org/abs/1908.10084.

Jie Ren, Jiaming Luo, Yao Zhao, Kundan Krishna, Mohammad Saleh, Balaji Lakshminarayanan, and Peter J Liu. Out-of-distribution detection and selective generation for conditional language models. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=kJUS5nD0vPB.

Pengzhen Ren, Yun Xiao, Xiaojun Chang, Po-Yao Huang, Zhihui Li, Brij B. Gupta, Xiaojiang Chen, and Xin Wang. A survey of deep active learning. *ACM Comput. Surv.*, 54(9), oct 2021. ISSN 0360-0300. doi:10.1145/3472291. URL https://doi.org/10.1145/3472291.

G. Riccardi and D. Hakkani-Tur. Active learning: theory and applications to automatic speech recognition. *IEEE Transactions on Speech and Audio Processing*, 13(4):504–511, 2005. doi:10.1109/TSA.2005.848882. URL https://ieeexplore.ieee.org/document/1453593.

Yuji Roh, Geon Heo, and Steven Euijong Whang. A survey on data collection for machine learning: A big data - ai integration perspective. *IEEE Transactions on Knowledge and Data Engineering*, 33(4):1328–1347, 2021. doi:10.1109/TKDE.2019.2946162. URL https://ieeexplore.ieee.org/document/8862913.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *ArXiv*, abs/1910.01108, 2019. URL https://api.semanticscholar.org/CorpusID:203626972.

Christopher Schröder, Lydia Müller, Andreas Niekler, and Martin Potthast. Small-text: Active learning for text classification in python, 2021.

Tal Schuster, Adam Fisch, and Regina Barzilay. Get your vitamin C! robust fact verification with contrastive evidence. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 624–643, Online, June 2021. Association for Computational Linguistics. doi:10.18653/v1/2021.naacl-main.52. URL https://aclanthology.org/2021.naacl-main.52.

Tal Schuster, Sihao Chen, Senaka Buthpitiya, Alex Fabrikant, and Donald Metzler. Stretching sentence-pair NLI models to reason over long documents and clusters. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 394–412, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics. doi:10.18653/v1/2022.findings-emnlp.28. URL https://aclanthology.org/2022.findings-emnlp.28.

P. Sebastiani and H. Wynn. Maximum entropy sampling and optimal bayesian experimental design. *Journal of the Royal Statistical Society Series B*, 62:145–157, 2000. URL https://www.jstor.org/stable/2680683.

Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=H1aIuk-RW.

Neha Sengupta, Sunil Kumar Sahu, Bokang Jia, Satheesh Katipomu, Haonan Li, Fajri Koto, William Marshall, Gurpreet Gosal, Cynthia Liu, Zhiming Chen, Osama Mohammed Afzal, Samta Kamboj, Onkar Pandit, Rahul Pal, Lalit Pradhan, Zain Muhammad Mujahid, Massa Baali, Xudong Han, Sondos Mahmoud

Bsharat, Alham Fikri Aji, Zhiqiang Shen, Zhengzhong Liu, Natalia Vassilieva, Joel Hestness, Andy Hock, Andrew Feldman, Jonathan Lee, Andrew Jackson, Hector Xuguang Ren, Preslav Nakov, Timothy Baldwin, and Eric Xing. Jais and jais-chat: Arabic-centric foundation and instruction-tuned open generative large language models. *arXiv preprint arXiv:2308.16149*, 2023. URL https://arxiv.org/abs/2308.16149.

Burr Settles, Mark Craven, and Soumya Ray. Multiple-instance active learning. In J. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems*, volume 20. Curran Associates, Inc., 2007. URL https://proceedings.neurips.cc/paper_files/paper/2007/file/a1519de5b5d44b31a01de013b9b51a80-Paper.pdf.

Artem Shelmanov, Vadim Liventsev, Danil Kireev, Nikita Khromov, Alexander Panchenko, Irina Fedulova, and Dmitry V. Dylov. Active learning with deep pre-trained models for sequence tagging of clinical and biomedical texts. In *2019 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pages 482–489, 2019. doi:10.1109/BIBM47256.2019.8983157. URL https://ieeexplore.ieee.org/document/8983157.

Yanyao Shen, Hyokun Yun, Zachary Lipton, Yakov Kronrod, and Animashree Anandkumar. Deep active learning for named entity recognition. In *Proceedings of the 2nd Workshop on Representation Learning for NLP*, pages 252–256, Vancouver, Canada, August 2017. Association for Computational Linguistics. doi:10.18653/v1/W17-2630. URL https://aclanthology.org/W17-2630.

Sunil Srinivasa. A review on multivariate mutual information. *University of Notre Dame, Indiana*, 2005. URL https://api.semanticscholar.org/CorpusID:7113625.

Jiao Sun, Yufei Tian, Wangchunshu Zhou, Nan Xu, Qian Hu, Rahul Gupta, John Wieting, Nanyun Peng, and Xuezhe Ma. Evaluating large language models on controlled generation tasks. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 3155–3168. Association for Computational Linguistics, December 2023a. doi:10.18653/v1/2023.emnlp-main.190. URL https://aclanthology.org/2023.emnlp-main.190.

Weiwei Sun, Lingyong Yan, Xinyu Ma, Shuaiqiang Wang, Pengjie Ren, Zhumin Chen, Dawei Yin, and Zhaochun Ren. Is ChatGPT good at search? investigating large language models as re-ranking agents. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 14918–14937, Singapore, December 2023b. Association for Computational Linguistics. doi:10.18653/v1/2023.emnlp-main.923. URL https://aclanthology.org/2023.emnlp-main.923.

Junya Takayama and Yuki Arase. Relevant and informative response generation using pointwise mutual information. In *Proceedings of the First Workshop on NLP for Conversational AI*, pages 133–138, Florence, Italy, August 2019.

Association for Computational Linguistics. doi:10.18653/v1/W19-4115. URL https://aclanthology.org/W19-4115.

Aleksandr Rubashevskii, Daria Kotova, and Maxim Panov. Scalable batch acquisition for deep bayesian active learning. In *Proceedings of the 2023 SIAM International Conference on Data Mining (SDM)*, pages 739–747, Minneapolis, Minnesota, U.S., April 2023. Society for Industrial and Applied Mathematics. doi:10.1137/1.9781611977653.ch83. URL https://epubs.siam.org/doi/abs/10.1137/1.9781611977653.ch83.

Arun James Thirunavukarasu, Darren Shu Jeng Ting, Kabilan Elangovan, Laura Gutierrez, Ting Fang Tan, and Daniel Shu Wei Ting. Large language models in medicine. *Nature medicine*, 29(8):1930–1940, 2023. URL https://www.nature.com/articles/s41591-023-02448-8.

James Thorne and Andreas Vlachos. Evidence-based factual error correction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3298–3309, Online, August 2021. Association for Computational Linguistics. doi:10.18653/v1/2021.acl-long.256. URL https://aclanthology.org/2021.acl-long.256.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. FEVER: a large-scale dataset for fact extraction and VERification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 809–819, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi:10.18653/v1/N18-1074. URL https://aclanthology.org/N18-1074.

Akim Tsvigun, Leonid Sanochkin, Daniil Larionov, Gleb Kuzmin, Artem Vazhentsev, Ivan Lazichny, Nikita Khromov, Danil Kireev, **Aleksandr Rubashevskii**, Olga Shahmatova, Dmitry V. Dylov, Igor Galitskiy, and Artem Shelmanov. ALToolbox: A set of tools for active learning annotation of natural language texts. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 406–434, Abu Dhabi, UAE, December 2022. Association for Computational Linguistics. doi:10.18653/v1/2022.emnlp-demos.41. URL https://aclanthology.org/2022.emnlp-demos.41.

E. Tsymbalov, M. Panov, and A. Shapeev. Dropout-based active learning for regression. In *nalysis of Images, Social Networks and Texts: 7th International Conference, AIST 2018*, pages 247–258. Springer, 2018. doi:10.1007/978-3-030-11027-7_24. URL https://doi.org/10.1007/978-3-030-11027-7_24.

Evgenii Tsymbalov, Sergei Makarychev, Alexander Shapeev, and Maxim Panov. Deeper connections between neural networks and gaussian processes speed-up active learning. In *IJCAI'19*, page 3599–3605. AAAI Press, 2019. URL https://www.ijcai.org/proceedings/2019/0499.pdf.

Liam van der Poel, Ryan Cotterell, and Clara Meister. Mutual information alleviates hallucinations in abstractive summarization. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 5956–5965. Association for Computational Linguistics, 2022. doi:10.18653/v1/2022.emnlp-main.399. URL https://aclanthology.org/2022.emnlp-main.399.

Minh Van Nguyen, Nghia Trung Ngo, Bonan Min, and Thien Huu Nguyen. Famie: A fast active learning framework for multilingual information extraction, 2022. URL https://arxiv.org/abs/2202.08316.

Neeraj Varshney, Wenlin Yao, Hongming Zhang, Jianshu Chen, and Dong Yu. A stitch in time saves nine: Detecting and mitigating hallucinations of llms by validating low-confidence generation. *arXiv preprint arXiv:2307.03987*, 2023. URL https://arxiv.org/abs/2307.03987.

Artem Vazhentsev, Gleb Kuzmin, Artem Shelmanov, Akim Tsvigun, Evgenii Tsymbalov, Kirill Fedyanin, Maxim Panov, Alexander Panchenko, Gleb Gusev, Mikhail Burtsev, Manvel Avetisian, and Leonid Zhukov. Uncertainty estimation of transformer predictions for misclassification detection. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8237–8252, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi:10.18653/v1/2022.acl-long.566. URL https://aclanthology.org/2022.acl-long.566.

Artem Vazhentsev, Gleb Kuzmin, Akim Tsvigun, Alexander Panchenko, Maxim Panov, Mikhail Burtsev, and Artem Shelmanov. Hybrid uncertainty quantification for selective text classification in ambiguous tasks. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11659–11681, Toronto, Canada, July 2023a. Association for Computational Linguistics. doi:10.18653/v1/2023.acl-long.652. URL https://aclanthology.org/2023.acl-long.652.

Artem Vazhentsev, Akim Tsvigun, Roman Vashurin, Sergey Petrakov, Daniil Vasilev, Maxim Panov, Alexander Panchenko, and Artem Shelmanov. Efficient out-of-domain detection for sequence to sequence models. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 1430–1454, Toronto, Canada, July 2023b. Association for Computational Linguistics. doi:10.18653/v1/2023.findings-acl.93. URL https://aclanthology.org/2023.findings-acl.93.

Carl Vondrick and Deva Ramanan. Video annotation and tracking with active learning. In J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 24. Curran Associates, Inc., 2011. URL https://proceedings.neurips.cc/paper_files/paper/2011/file/182be0c5cdcd5072bb1864cdee4d3d6e-Paper.pdf.

Tu Vu, Mohit Iyyer, Xuezhi Wang, Noah Constant, Jerry Wei, Jason Wei, Chris Tar, Yun-Hsuan Sung, Denny Zhou, Quoc Le, et al. Freshllms: Refreshing large language models with search engine augmentation. *arXiv preprint arXiv:2310.03214*, 2023. URL https://arxiv.org/pdf/2310.03214.pdf.

David Wadden, Shanchuan Lin, Kyle Lo, Lucy Lu Wang, Madeleine van Zuylen, Arman Cohan, and Hannaneh Hajishirzi. Fact or fiction: Verifying scientific claims. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7534–7550, Online, November 2020. Association for Computational Linguistics. doi:10.18653/v1/2020.emnlp-main.609. URL https://aclanthology.org/2020.emnlp-main.609.

Fang Wan, Tianning Yuan, Mengying Fu, Xiangyang Ji, Qingming Huang, and Qixiang Ye. Nearest neighbor classifier embedded network for active learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 10041–10048, 2021. URL https://ojs.aaai.org/index.php/AAAI/article/view/17205.

Hao Wang, Yuxia Wang, Minghan Wang, Yilin Geng, Zhen Zhao, Zenan Zhai, Preslav Nakov, Timothy Baldwin, Xudong Han, and Haonan Li. Loki: An open-source tool for fact verification, 04 2024a. URL https://zenodo.org/records/11004461.

Xilu Wang, Yaochu Jin, Sebastian Schmitt, and Markus Olhofer. Recent advances in bayesian optimization. *ACM Comput. Surv.*, 55(13s), jul 2023. ISSN 0360-0300. doi:10.1145/3582078. URL https://doi.org/10.1145/3582078.

Yuxia Wang, Revanth Gangi Reddy, Zain Muhammad Mujahid, Arnav Arora, **Aleksandr Rubashevskii**, Jiahui Geng, Osama Mohammed Afzal, Liangming Pan, Nadav Borenstein, Aditya Pillai, Isabelle Augenstein, Iryna Gurevych, and Preslav Nakov. Factcheck-bench: Fine-grained evaluation benchmark for automatic fact-checkers. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 14199–14230, Miami, Florida, USA, November 2024b. Association for Computational Linguistics. doi:10.18653/v1/2024.findings-emnlp.830. URL https://aclanthology.org/2024.findings-emnlp.830.

Yuxia Wang, Revanth Gangi Reddy, Zain Muhammad Mujahid, Arnav Arora, Aleksandr Rubashevskii, Jiahui Geng, Osama Mohammed Afzal, Liangming Pan, Nadav Borenstein, Aditya Pillai, Isabelle Augenstein, Iryna Gurevych, and Preslav Nakov. Factcheck-bench: Fine-grained evaluation benchmark for automatic fact-checkers. *arXiv preprint arXiv:2311.09000*, 2024c. URL https://arxiv.org/abs/2311.09000.

Yuxia Wang, Minghan Wang, Muhammad Arslan Manzoor, Georgi Georgiev, Rocktim Jyoti Das, and Preslav Nakov. Factuality of large language models in the year 2024. *CoRR*, abs/2402.02420, 2024d. doi:10.48550/ARXIV.2402.02420. URL https://doi.org/10.48550/arXiv.2402.02420.

Alex Warstadt, Amanpreet Singh, and Samuel R. Bowman. Neural Network Acceptability Judgments. *Transactions of the Association for Computational Linguistics*, 7:625–641, 09 2019. doi:10.1162/tacl_a_00290. URL https://doi.org/10.1162/tacl_a_00290.

Satosi Watanabe. Information theoretical analysis of multivariate correlation. *IBM Journal of Research and Development*, 4(1):66–82, 1960. doi:10.1147/rd.41.0066. URL https://ieeexplore.ieee.org/document/5392532.

Max Wiechmann, Seid Muhie Yimam, and Chris Biemann. ActiveAnno: General-purpose document-level annotation tool with active learning integration. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Demonstrations*, pages 99–105, Online, June 2021. Association for Computational Linguistics. doi:10.18653/v1/2021.naacl-demos.12. URL https://aclanthology.org/2021.naacl-demos.12.

Dustin Wright and Isabelle Augenstein. Claim check-worthiness detection as positive unlabelled learning. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 476–488, Online, November 2020. Association for Computational Linguistics. doi:10.18653/v1/2020.findings-emnlp.43. URL https://aclanthology.org/2020.findings-emnlp.43.

Dustin Wright, David Wadden, Kyle Lo, Bailey Kuehl, Arman Cohan, Isabelle Augenstein, and Lucy Lu Wang. Generating scientific claims for zero-shot scientific fact checking. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2448–2460, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi:10.18653/v1/2022.acl-long.175. URL https://aclanthology.org/2022.acl-long.175.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *CoRR*, 1708.07747, 2017. URL https://arxiv.org/abs/1708.07747.

Jingkang Yang, Kaiyang Zhou, Yixuan Li, and Ziwei Liu. Generalized out-of-distribution detection: A survey. *International Journal of Computer Vision*, pages 1–28, 2024.

Yao-Yuan Yang, Shao-Chuan Lee, Yu-An Chung, Tung-En Wu, Si-An Chen, and Hsuan-Tien Lin. libact: Pool-based active learning in python. *arXiv preprint arXiv:1710.00379*, 2017.

Zhangyue Yin, Qiushi Sun, Qipeng Guo, Jiawen Wu, Xipeng Qiu, and Xuanjing Huang. Do large language models know what they don't know? In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 8653–8665, Toronto, Canada, July 2023. Association for Computational Linguistics. doi:10.18653/v1/2023.findings-acl.551. URL https://aclanthology.org/2023.findings-acl.551.

01.AI: Alex Young, Bei Chen, Chao Li, Chengen Huang, Ge Zhang, Guanwei Zhang, Heng Li, Jiangcheng Zhu, Jianqun Chen, Jing Chang, Kaidong Yu, Peng Liu, Qiang Liu, Shawn Yue, Senbin Yang, Shiming Yang, Tao Yu, Wen Xie, Wenhao Huang, Xiaohui Hu, Xiaoyi Ren, Xinyao Niu, Pengcheng Nie, Yuchi Xu, Yudong Liu, Yue Wang, Yuxuan Cai, Zhenyu Gu, Zhiyuan Liu, and Zonghong Dai. Yi: Open foundation models by 01.ai, 2024. URL https://arxiv.org/abs/2403.04652.

Michelle Yuan, Hsuan-Tien Lin, and Jordan Boyd-Graber. Cold-start active learning through self-supervised language modeling. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages

7935–7948, Online, November 2020. Association for Computational Linguistics. doi:10.18653/v1/2020.emnlp-main.637. URL https://aclanthology.org/2020.emnlp-main.637.

Muru Zhang, Ofir Press, William Merrill, Alisa Liu, and Noah A. Smith. How language model hallucinations can snowball. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=FPlaQyAGHu.

Xiang Zhang, Junbo Jake Zhao, and Yann LeCun. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 649–657, 2015. URL https://proceedings.neurips.cc/paper/2015/hash/250cf8b51c773f3f8dc8b4be867a9a02-Abstract.html.

Yue Zhang, Yafu Li, Leyang Cui, Deng Cai, Lemao Liu, Tingchen Fu, Xinting Huang, Enbo Zhao, Yu Zhang, Yulong Chen, Longyue Wang, Anh Tuan Luu, Wei Bi, Freda Shi, and Shuming Shi. Siren's song in the AI ocean: A survey on hallucination in large language models. *CoRR*, abs/2309.01219, 2023. doi:10.48550/arXiv.2309.01219. URL https://doi.org/10.48550/arXiv.2309.01219.

Guang Zhao, Edward Dougherty, Byung-Jun Yoon, Francis J. Alexander, and Xiaoning Qian. Bayesian active learning by soft mean objective cost of uncertainty. In *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*, volume 130 of *Proceedings of Machine Learning Research*, pages 3970–3978. PMLR, 13–15 Apr 2021. URL https://proceedings.mlr.press/v130/zhao21c.html.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in Neural Information Processing Systems*, 36, 2024.

Appendix A

Additional active learning results

A.1 Additional results for text data

In addition, we provide figures for AG news dataset, collecting all the considered algorithms, including also the Least Confidence (LC) method and the current SOTA algorithm BADGE, in which additional clustering provides diversity of selected samples. As we can see, our algorithm shows a top-performance comparable to BADGE for both smaller (10) and larger (50) batch sizes, see Figure A-1 and Figure A-2, respectively. At the same time, LBB shows higher quality compared to BALD and BatchBALD on a small batch, while losing the quality advantage when running the algorithm on a larger batch. This may be due to some performance saturation without additional randomization, or the difficulty of dealing with a small number of classes in the presence of a large amount of data.

As a generalization, we analyze the effectiveness of the proposed method compared to alternatives using the so-called Dolan-More curves, which we will discuss next.

A.2 MC-dropout results

Comparing the results obtained with MC-dropout and with deep ensembles, we see that the test accuracy of algorithms with MC-dropout is lower than with ensembles, as we mentioned earlier, see Figure A-3a, Figure A-3b, Figure A-3c. At the same time, the margin between algorithms, for example, between LBB and BALD is

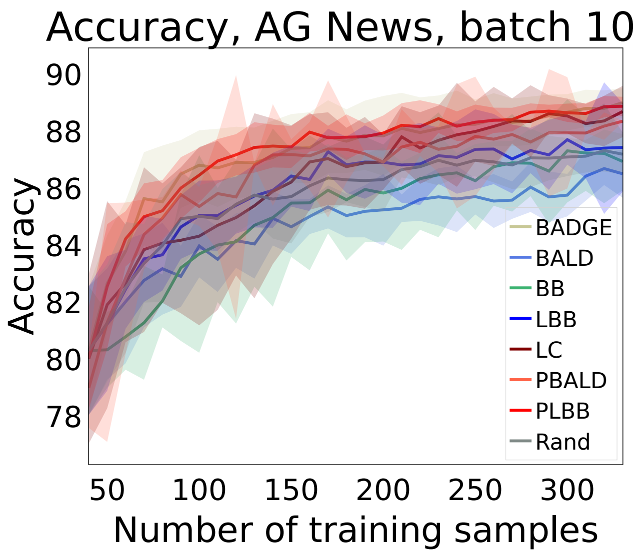


Figure A-1: Performance comparison of AL algorithms on AG News dataset with a batch 10. PLBB and BADGE have comparable top-performance.

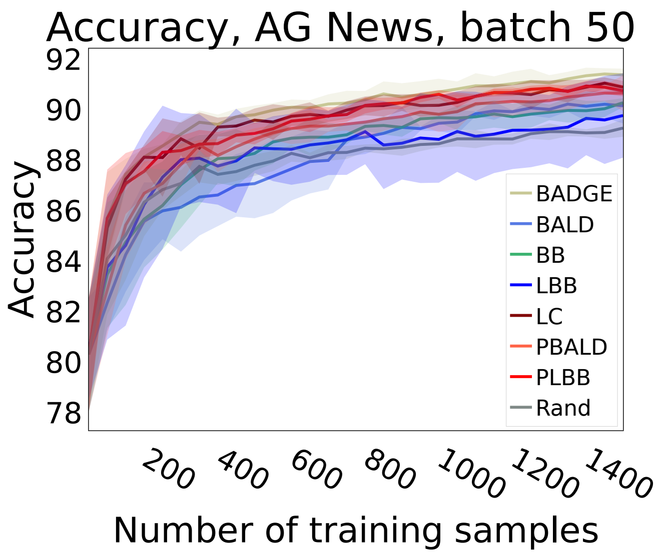
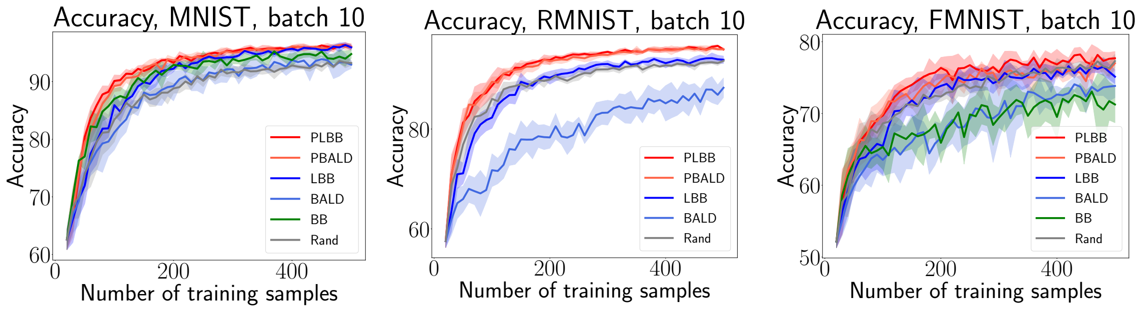


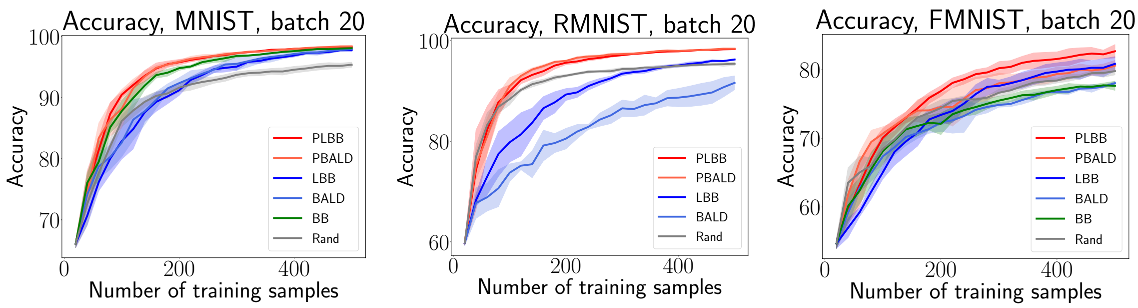
Figure A-2: Performance comparison of AL algorithms on AG News dataset with a batch 50. PLBB and BADGE are two top-performers among alternatives.

more clear in the figures obtained with MC-dropout. Thus, the Large BatchBALD algorithm is significantly better than BALD. Also, in Figure A-3a the LBB algorithm even outperforms BatchBALD in quality, starting from 200 elements in the training set. The BALD algorithm, however, shows quality comparable to random selection of samples. Furthermore, in Figure A-3b and Figure A-3c the BALD performance is significantly worse than the random selection of samples. Also, in all three figures, power extensions show the best accuracy among all algorithms.



(a) (b) (c)

Figure A-3: Performance comparison of AL algorithms on MC-dropout. Datasets: (a) MNIST. (b) RMNIST. (c) FMNIST. LBB is clearly superior to BALD, and by a larger margin than in the case of deep ensembles.



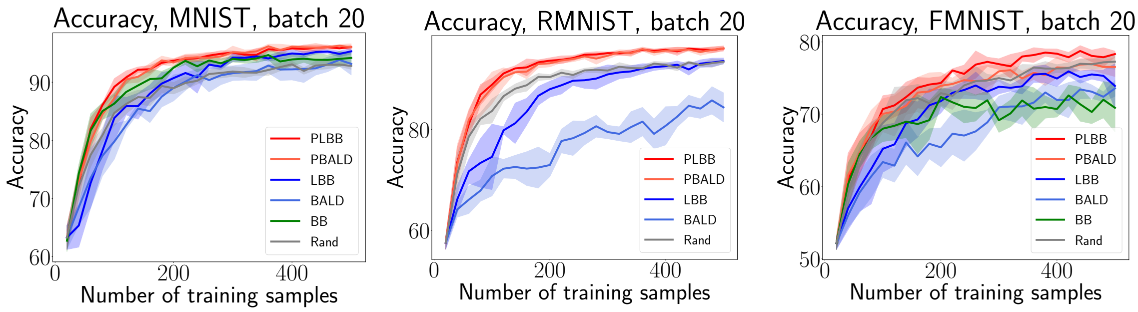
(a) (b) (c)

Figure A-4: Performance comparison of AL algorithms on deep ensembles. Datasets: (a) MNIST. (b) RMNIST. (c) FMNIST.

A.3 Additional results for bigger batches

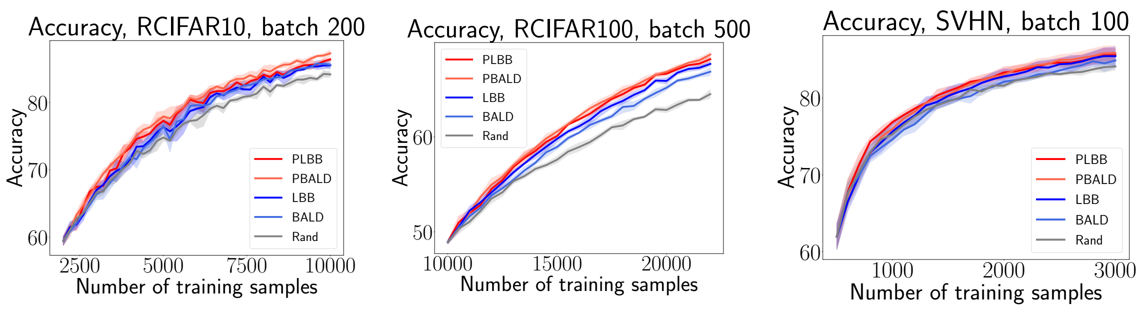
Experimental results on a larger acquisition batch for the MNIST dataset are shown in Figure A-4a. Note that LBB and BALD have comparable accuracy here, as do PLBB and PBALD. Power extensions also dominate among other algorithms. The results for the RMNIST dataset with larger acquisition batch size are in Figure A-4b. Here we observe a similar picture for LBB and BALD, as well as for PLBB and PBALD, which in turn show the best quality among all algorithms. Similar conclusions can be drawn from the FMNIST dataset, with acquisition batch size equal to 20, see Figure A-4c. For results on the MNIST, RMNIST, FMNIST datasets on the 20 batch with MC-dropout, see Figure A-5a, Figure A-5b, Figure A-5c, respectively.

Regarding the SVHN dataset with a batch size of 100 samples, see Figure A-6c, Large BatchBALD shows better quality compared to BALD. In turn, BALD has a



(a) (b) (c)

Figure A-5: Performance comparison of AL algorithms on MC-dropout. Datasets: (a) MNIST. (b) RMNIST. (c) FMNIST.



(a) (b) (c)

Figure A-6: Performance comparison of AL algorithms on deep ensembles. Datasets: (a) RCIFAR-10. (b) RCIFAR-100. (c) SVHN.

similar performance to the random baseline. In the same figure, the PLBB algorithm slightly outperforms PBALD on the first few thousand elements, after which they have similar quality superior to the competitors. The results for RCIFAR-10 on a batch size 200 are shown in Figure A-6a. Also, on RCIFAR-100 with a larger batch the Large BatchBALD algorithm outperforms the BALD algorithm, see Figure A-6b.