DELFT UNIVERSITY OF TECHNOLOGY

Masters Thesis

Blind Spot Illumination in LLMs through Data Valuation and Synthetic Sample Generation

Author:
C.C.I.A. Chen

Student number
4725204

Committee Chair:
Dr. A. Katsifodimos

Daily Supervisor
Dr. J. Yang

External Member
Dr. Q. Wang

Daily Co-Supervisor:
P. Lippmann

To obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on 22 April, 2024 at 11:00

An electronic version of this thesis is available at http://repository.tudelft.nl/.

This page was intentionally left blank

Abstract

Large language models (LMs) are increasingly used in critical tasks, making it important that these models can be trusted. The confidence an LM assigns to its prediction is often used to indicate how much trust can be placed in that prediction. However, a high confidence can be incorrectly trusted if it turns out to be incorrect, also known as a high-confident error, or *unknown unknown* (UU). Blind spots, clusters of UUs, are caused by out-of-distribution data (OOD), bias, covariate shift, or unseen data. Previous work by Lippmann et al. generated samples based on these UUs, however, they selected a random subset of all found UUs. Although this was the first study that could mitigate blind spots without the need for crowd workers or oracles, they did not address the underlying causes of blind spots, such as OOD and bias. Data valuation is the research field that is concerned with valuing samples to improve model performance and can identify OOD data, bias, and noise. This paper proposed to combine the research field of data valuation, with the mitigation of blind spots. We generated synthetic samples using the highest-valued samples and retrained the LM using a Weighted Loss based on Data Values (WLDV). We conducted an extensive evaluation of our approach on four tasks, demonstrating a reduction of UUs by up to 32.7%, while retaining the same level of accuracy. This was the first exploration of combining the research fields of data valuation and blind spot mitigation.

Acknowledgments

This work would not have been possible without the help Philip Lippmann and dr. Jie Yang. I really appreciate the guidance and support throughout the project. I am deeply thankful for the support and guidance provided by Philip Lippmann, the weekly discussions we held were very insightful for my process. I would also like to thanks to the Delft University of Technology for their resources and financial support for conducting the experiments. I would also like to thank my thesis committee, dr. Qing Wang and dr. Asterios Katsifodimos for reviewing the study. Finally, I would like to express my most profound appreciation to my family and friends.

List of abbreviations

Abbreviation	Fully written
UU(s)	**U**nknown **u**nknown(s)
BERT	**b**idirectional **e**ncoder **r**epresentations from **t**ransformers
LMs	**l**anguage **m**odels
LLMs	**l**arge **l**anguage **m**odels
DV	**d**ata **v**aluation
DVRL	**D**ata **v**aluation using **R**einforcement **l**earning.
OOD	**O**ut of **d**istribution **d**ata.
ECE	**e**stimated **c**alibration **e**rror
BLISS	Blind Spot Illumination in LLMs through Data Valuation and Synthetic Sample Generation
IBS	**I**lluminating **B**lind **S**pots by Leveraging Humans and Language Models

Contents

Introduction

The confidence large Language Models (LMs) assign to their predictions, is often used to indicate how much trust can be placed in a prediction. A high-confident prediction can be trusted over a low-confident prediction. If a high-confident prediction is incorrect, this can lead to critical mistakes. Especially as LMs are increasingly adopted in various domains. For example, if an LM classifies a medical diagnosis incorrectly and assigns it high confidence, a doctor may trust the prediction. This can lead to the patient being ignored by subsequent human inspection and cause an irreparable error. A low-confidence prediction, on the other hand, may not be relied upon and sent for additional review. It is therefore important to identify and mitigate these high-confident incorrect predictions, which are called unknown unknowns (UUs). The challenge lies in the models' inability to identify their own shortcomings, as it is inherently confident about their own prediction. However, previous work found that UUs tend to cluster together (Attenberg, Ipeirotis, and Provost 2015). This cluster is called a blind spot. The goal of this study is to improve the trustworthiness of LMs by mitigating blind spots, applied to text classification problems.

1.1. Problem Statement

Previous studies have mostly been concerned about discovering and mitigating blind spots using not-scalable methods, such as oracles or crowd workers (Lakkaraju et al. 2017; Bansal and Weld 2018; Attenberg, Ipeirotis, and Provost 2015). Mitigating blind spots without crowd workers is the next step that can be taken. The study of Lippmann et al. extended the training set with synthetically generated samples based on UUs and found that this reduced blind spots (Lippmann, Spaan, and Yang 2024). They compared the effectiveness of LMs and humans in generating synthetic samples and found that LMs outperform humans in characterizing blind spots. This approach facilitated a new cost-effective way to mitigate blind spots.

In the literature, there is a consensus that blind spots are caused by out-of-distribution (OOD) data, bias, covariate shift, or unseen data (Chung et al. 2019; Attenberg, Ipeirotis, and Provost 2015). These concepts are similar and indicate that the training data may have a different distribution compared to the distribution seen at run time. E.g., when training a classification model to predict a dog's race trained on only mature dog images, it is understandable that the model gives more incorrect predictions when showing puppy images. The prediction would be a UU if the model predicts there is a high chance that the puppy is a 'Labrador' if, in reality, it would be a different race. The research field of data valuation (DV) is concerned about valuing these samples (Sim, Xu, and Low 2022). The value they attribute to each sample in the training set indicates the importance of the sample to the model's performance. These DV methods can be used in OOD scenarios to improve the model's performance. We, therefore, hypothesize that DV, in combination with synthetic sample generation, could improve existing blind spot mitigation methods. Hence, the objective of this study is to answer the research question:

RQ How can data generation and data valuation be used to mitigate blind spots in large language models?

- **SQ1**: How can data values be integrated during the retraining of the LM?
- **SQ2**: How to select UUs to generate synthetic samples that reduce blind spots?
- **SQ3**: What is the effect when combining the optimized selection procedure for synthetic sample generation, and then retraining the model using data values?

1.2. The work

We investigate two research directions to incorporate data valuation to mitigate blind spots. First, data valuation is used as a selection procedure to select which training samples should used to generate new synthetic samples. Second, introducing a Weighted Loss based on Data Values (WLDV) when retraining the LM.

The first exploration extends the research of Lippmann et al. who generated synthetic samples based on UUs (Lippmann, Spaan, and Yang 2024). To elicit these UU predictions, they perturbed input sentences using the adversarial attack methods DeepWordBug or TextFooler. Due to the abundance of UUs created after the attack, they randomly selected a subset of these UUs and used the subset to generate the synthetic samples. We extended their approach by, instead of random sampling UUs, targeting only high-valued samples. We hypothesize that this is more effective in blind spot mitigation since these high-valued samples may represent OOD data, bias, or underrepresented areas (Sim, Xu, and Low 2022; Yoon, Arik, and Pfister 2020), which are assumed to be the causes of blind spots (Chung et al. 2019; Attenberg, Ipeirotis, and Provost 2015). There exist multiple methods to calculate the data values. In this study, we will calculate the values using the state-of-the-art method Data Valuation using Reinforcement Learning (DVRL) by Yoon et al. (Yoon, Arik, and Pfister 2020). Their method can calculate the data values for each sample. To measure the effectiveness, a comparative analysis will be conducted between synthetic samples generated from randomly chosen UUs, referred to as ”Illuminating Blind Spots: Exploring LLMs as a Source of Targeted Synthetic Textual Data to Minimize High Confidence Misclassifications” (IBS) and those derived from high-valued UUs, referred to as ”Blind Spot Illumination using Data Valuation and Synthetic Sample Generation” (BLISS). In Figure 1.1 this optimization is depicted as the first two steps in the pipeline.

The second exploration introduces data valuation when retraining the LM. To do this, we extend the DVRL method. The DVRL method consists of two models: a data value estimator (DVE), and a predictor model. The DVE estimates the data values and learns this jointly with the predictor model using a reinforcement learning signal. It does this in iterations. Their method scales to large datasets, but only if each iteration is fast. In this study we use the increasingly popular LM as a predictor model, however, using an LM would imply retraining the LM every iteration. However, an LM is computationally complex to retrain, which will make each iteration too slow and make the system unusable. We address this issue by employing a basic predictor model that is quick to retrain. To still keep the contextual knowledge of the LM, the basic predictor model uses text embeddings as its input. Due to its simplicity, the model isn't used in downstream tasks. Instead, we compute the data values through the DVRL method with this basic predictor. The calculated data values are then used to retrain the LM once, applying a Weighted Loss based on those Data Values (WLDV). This approach has a drawback in that it does not enable concurrent learning between the data values and the predictor model. However, we address this by retraining the model after calculating the data values, using these data values. This allows the LM to learn from important samples, and give less emphasis on unimportant samples. In figure 1.1 the retraining step is depicted as the bottom part of the pipeline.

The model has been evaluated on 4 distinct text classification tasks, which each 6 different scenarios. The findings indicate that random sampling UUs from the perturbed training set, in contrast to our hypothesis, outperforms selecting UUs derived from high-valued samples. However, incorporating a weighted loss based on data values during the model's fine-tuning process enhances overall model performance. The best performance was therefore found by first generating synthetic samples based on random sampled UUs and then retraining the LM using WLDV.

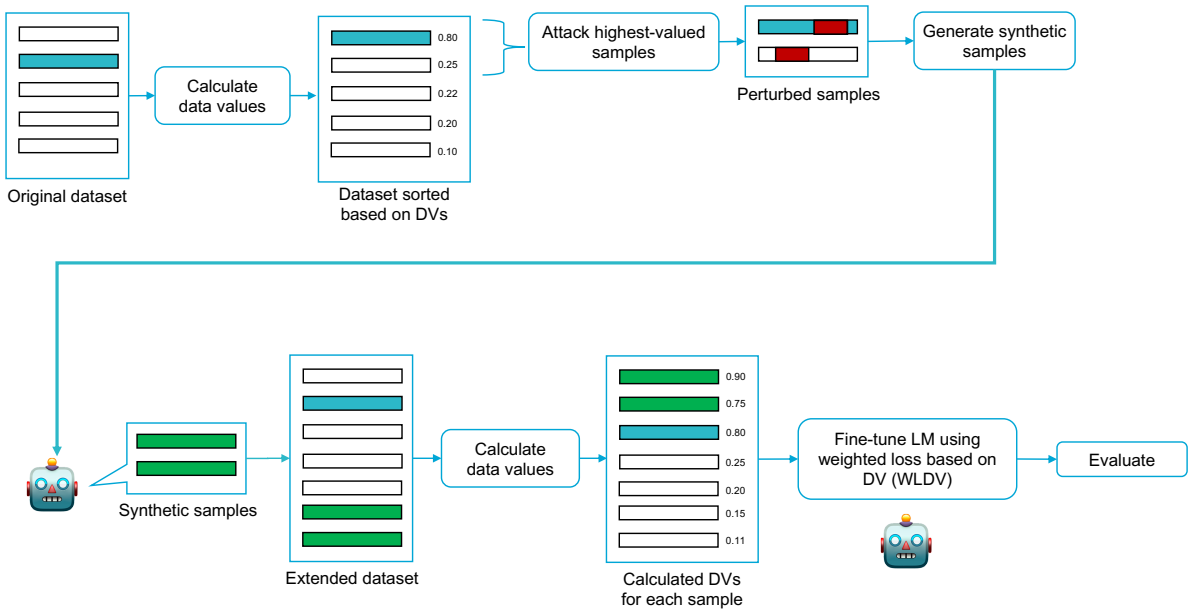


Figure 1.1: Pipeline of the proposed methodology. First, the samples are sorted based on the data values of each sample. Then, the highest-valued samples are attacked, creating UUs. Based on those UUs, synthetic samples are generated. After extending the training set with the synthetic samples, the data values are calculated again and the LM is fine-tuned using a weighted loss based on the data values that were calculated the second time (WLDV).

1.3. Contributions

Data valuation has, to the best of our knowledge, not been used before to mitigate blind spots in LM. Its importance and uniqueness lie in providing a clearer understanding of the connection between data values and the detection and mitigation of blind spots.

In summary, the present study will make the following contributions:

- Present a methodology to mitigate blind spots using data valuation in a scalable manner.
- Demonstrate that retraining a LM using weighted loss based on data valuation reduces blind spots.
- Demonstrate that synthetic samples generation based on random selected UUs, is preferred over a selection of UUs based on high data values.

The present study is divided into the following sections, first, the related study is presented in chapter 2. Then the methodology in chapter 3, the results in chapter 5. Finally, the discussion (chapter 6) and conclusion (chapter 7).

Related work

This section explores blind spots, adversarial attacks, and data valuation methods. Section 2.1 explains the challenge to discover/mitigate blind spots and discusses state-of-the-art solutions. Section 2.2 discusses adversarial attack methods, including some aimed at blind spots. In section 2.3 discusses data valuation methods and challenges.

2.1. Blind spots

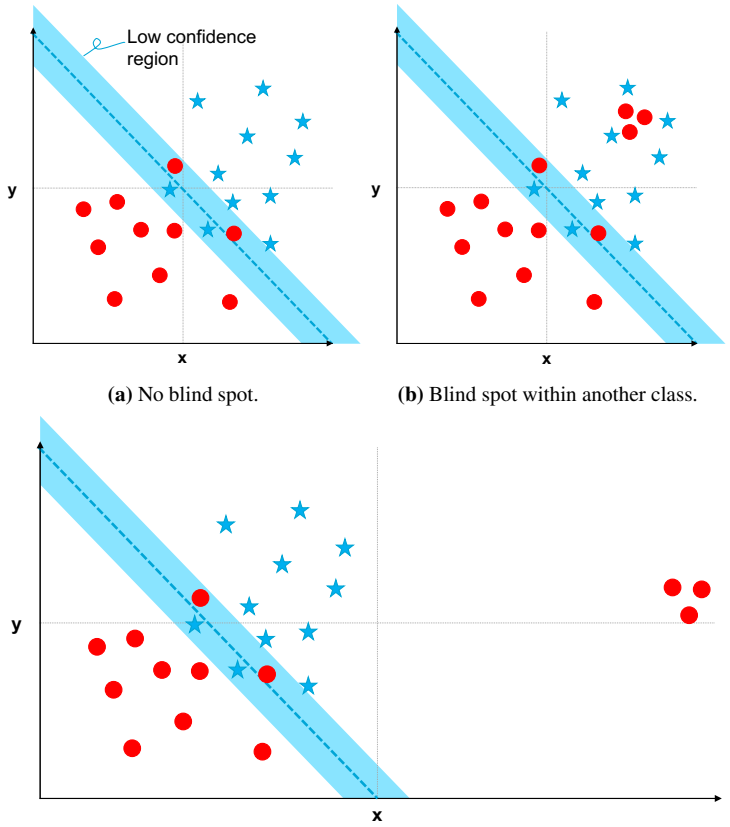
In this section, we will first discuss the definition of a blind spot (section 2.1.1). In the subsequent two sections, we will delve into the two classifications of blind spots that we have identified: crowd-sourcing algorithms (section 2.1.2) and generation-based approaches (section 2.1.3). We discuss evaluation metrics for blind spot mitigation in section 2.1.4, as there is no uniform way to measure it. At last, we discuss the related concept of probability calibration (section 2.1.4)

2.1.1. Definition

Before discussing the blind spot mitigation techniques, it is necessary to formalize the definition, as there is some ambiguity in the literature. In line with other studies (Lippmann, Spaan, and Yang 2024; Lakkaraju et al. 2017; Bansal and Weld 2018), this study defined blind spots as a cluster of unknown unknowns (UUs). Some use the term blind spot and UUs interchangeably (Cabrera et al. 2021), while others use the term 'unknown unknown' to indicate a test sample that is unseen during training time (Chung et al. 2019). This is not a precise definition, but it is intended to convey a more informal understanding of the relationship between UUs and blind spots.

An unknown unknown (UU) prediction, is a highly confident prediction that is incorrect. More formally, a predictor model θ predicts a label $\hat{y}$ for sample x with confidence c higher than τ, but the prediction was incorrect $\hat{y} \neq y$. Inline with previous studies, we assign $\tau \geq 0.65$ (Lakkaraju et al. 2017; Bansal and Weld 2018; Lippmann, Spaan, and Yang 2024). A more elaborated definition is given in section 3.1, Definition 3.1.

Blind spots can occur when the training distribution D is different compared to the target, or deployment, distribution T. This difference can occur due to any reason, such as bias, class imbalance, or time-related effects. Although $D \neq T$, we still assume that the probability of sampling from D, $p_D(x|y)$, is equal to sampling from T, $p_T(x|y)$. This can cause UU predictions (Chung et al. 2019). Attenberg et al found that UUs cluster systematically in regions of the feature space, and are not just outliers (Attenberg, Ipeirotis, and Provost 2015). Figures 2.1b and 2.1c represent two types of blind spots. The first blind spot occurs within the distribution of another class, while the other blind spot occurs in an unseen area.



(a) No blind spot.

(b) Blind spot within another class.

(c) Blind spot in OOD feature space.

Figure 2.1: The figures show two different types of blind spots (b) and (c). The further away samples are from the decision boundary (dashed line) the higher the confidence of the prediction. The padding surrounding the decision boundary is a low confidence region. Figure a shows a normal distribution, b shows a blind spot in the middle of the other distribution. Figure c shows a blind spot in unseen area.

2.1.2. Crowdsourcing

This section examines research that uses human oversight as part of their methodology. These studies do not scale well to large systems. Nonetheless, some subsequent studies built upon these foundational ideas and attempted to address the issue of human dependence. Understanding their problems and solutions provides valuable insights to understand other studies.

The influential paper Beat The Machine (BTM) (Attenberg, Ipeirotis, and Provost 2015) proposed a system to generate new samples in the blind spot of a predictor model. They did this by leveraging crowd workers to generate samples. To incentivize the crowd workers to provide high-quality examples that were difficult for a model to predict, the reward was proportional to the magnitude of a predictor model's error. The study is limited in two ways. First, it required humans to fool the model, but also required a set of (trusted) humans to validate if the model was incorrect. The validation cannot be done automatically by the predictor model itself, as the purpose of the samples was that the model failed to predict the correct label. To partially overcome the requirement of the crowd workers, Lui et al. (Cabrera et al. 2021), extended the original BTM method and proposed a hybrid human-machine approach called Patterned Beat-The-Machine (P-BTM). They, similar to BTM, asked crowd workers to provide initial labels for UUs, but those samples were used to train a classifier to find more examples in an unlabeled dataset. Like clustering methods, P-BTM requires a classification model, which is often challenging. Additionally, it required labeling initial UUs using a mechanism like BTM.

Van der hof et al. proposed a solution to reduce the required human oversight in both selecting samples and classifying samples (Vandenhof 2019). They automated the process of selecting high

confident samples, by generating a set of interpretable decision rules that explained how the model predicted high confident predictions. Crowd workers were then challenged to adjust a sample such that one of the rules was contradicted, but the sample was still valid. Although they automated the selection of high-confidence samples, they could not fully eliminate crowd workers.

Similar to our study, Han et al., tried to improve under-represented classes and shift towards a more balanced training set (L. Han, Dong, and Demartini 2021). They compared the feature space created by crowd workers and the feature space learned by a predictor model. Based on the difference between these two feature spaces, they identified UUs (Figure 2.2). These UUs were sent to the crowd works to be labeled. Using the newly labeled samples, the original training set was extended and the model was retrained. After retraining, the model's feature space changed, allowing the identification of a different set of UUs. They claimed that their method is a cost-efficient way to discover and mitigate UUs. The method in this study does not utilize this selection method, instead utilizes data valuation techniques to identify interesting UUs.

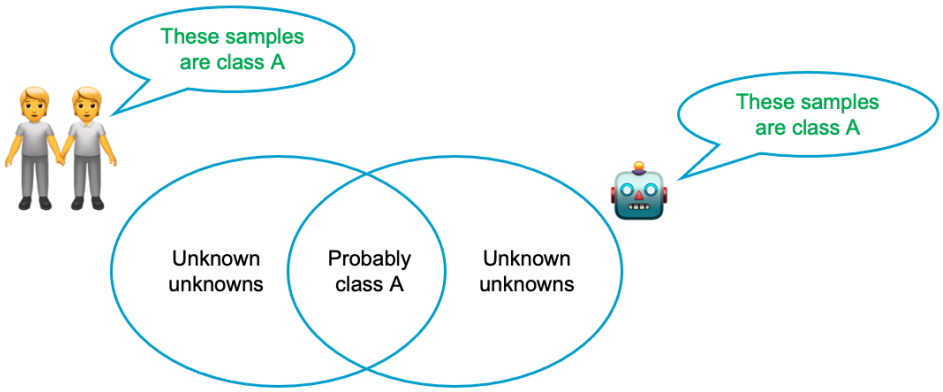


Figure 2.2: Illustration to explain Han et al.'s method (L. Han, Dong, and Demartini 2021). Both humans and LM assign different areas of the feature space to a class A. By looking at the difference between the two feature spaces, possible UUs can be identified and further investigated.

2.1.3. Unsupervised approaches

In the previous subsection, we discussed methodologies that required human supervision. This section discusses methodologies that do not require human intervention. To discover and mitigate UUs, Lakkaraju et al. proposed to select UUs as effectively as possible, by minimizing a cost (Lakkaraju et al. 2017). They introduced a two-step method. First, they used the fact that UUs cluster together and partitioned all the samples. The goal was to maximize the similarity between samples in the same partition and maximize the difference between the partitions. Then, they proposed a multi-armed bandit algorithm to select a partition and sample samples from that partition. The selected sample was shown to the oracle. The oracle was able to obtain the true label of the sample, verifying if the sample was a UU. Lakkaraju et al. introduced a high cost $cost(x(t))$ to query the oracle with sample x at timestep t. Therefore, the goal was to optimize which samples should be queried to the oracle to classify all unseen data correctly. If the queried sample was a UU, a reward of 1 was given, else no reward was given. Optimization was now formulated according to Equation 2.1, in which the utility should be maximized. The reward was indicated by $\mathbb{1}_{o(x_t)\neq c}$. A downside of the optimization is that each discovered UU gave the same reward, which does not support the discovery of new blind spots. Additionally, an oracle was required, which is commonly carried out by crowd workers. This still does not eliminate the need for humans.

$$u(x(t)) = \mathbb{1}_{\{o(x_t)\neq c\}} - \gamma \times \cos t(x(t)) \tag{2.1}$$

Bansal et al. observed that once a UU was found, new samples were selected that were near the UU. Since it is already previously known that UUs cluster together, finding new UUs near previously

found UUs does not add much additional information. They extended the reward function by adding a usefulness reward U (Equation 2.2), which includes the degree to which the selected sample has been explained by the already queried set Q (Equation 2.3). To calculate how much set Q explained the sampled sample x, they assumed a similarity function. By adjusting the reward function based on the dissimilarity of previously identified UUs, the model is encouraged to find UUs that are more spread out in the feature space. This improves the illumination of previous unknown blind spots and gives a better "coverage" of the feature space.

$$U_M\left(Q, y_Q\right) = \sum_{x \in D_{\text{test}}} c_{M,x} \cdot P\Big(\text{explain}_x \mid Q, y_Q\Big) \tag{2.2}$$

$$P\Big(\text{explain}_x \mid Q, y_Q\Big) := \max_{q \in Q} \mathbb{1}_S(q) \cdot \text{sim}(x, q) \tag{2.3}$$

While the algorithmic methods proposed by Lakkaraju et al. and Bansal et al. are effective for identifying UUs in a fixed test set, they may not be suitable for predictive models deployed in the open world. In such cases, models must be continually evaluated on new data, and identifying UUs in a fixed set may not be sufficient. Therefore, we aim to develop methods that can identify UUs encountered by models in a scenario in which the data may be evolving or changing.

The work of Lippmann et al. shed light on the use of synthetic data generation to address blind spots within LMs (Lippmann, Spaan, and Yang 2024). They compared human-generated synthetic samples to LM-generated synthetic samples and found that both decreased the number of UUs effectively. However, the LM-generated samples were more cost-effective, with 4,3% of the total cost. This opened a scalable solution for blind spot mitigation. The synthetic samples were based on UUs. They tried to elicit these UUs by perturbing the input sentences using the adversarial attack methods DeepWordBug (Gao et al. 2018) or TextFooler (Jin et al. 2020). The adversarial methods created an abundance of UUs and they, therefore, selected a random subset of UUs. For each UU, they created two distinct types of synthetic samples: one aimed at exploiting an existing blind spot and another focused on exploring a new, potentially undiscovered blind spot. To exploit an existing blind spot, an LM formulated a generative hypothesis that explained the reason(s) why the perturbation created a UU. To explore a new blind spot, they asked the LM to adapt the generative hypothesis. Using both hypotheses, two synthetic samples were generated. These synthetic samples were used to extend the training set and retrain the LM. Despite the promising results, the model's approach of selecting a random subset of UUs raises questions about the optimization of this selection process. The underlying causes of blind spots, such as OOD data, bias, or under-presented areas in the feature space were not addressed in their method. This study adopts their idea of synthetic sample generation, however, instead of randomly selecting UUs, we propose to use a selection mechanism that is based on data valuation.

The research conducted by Chung et al. explored the consequences of covariate shift (Chung et al. 2019). This occurs when the training and test data distributions differ, which can occur for any reason. Even well-trained models may struggle to generalize to the test set when a covariance shift is present. The challenge of learning under covariance shift has been extensively studied, however, most of those require a (limited) set of test samples (Chung et al. 2019). Chung et al proposed a technique in which this was not required. Their method used the training set, alongside data redundancy information. They hypothesized that a dataset constructed from multiple sources can contain redundant information. They used this redundant information, together with species-estimation techniques, to detect UUs and estimate missing unseen test instances U. Similar to our study, they corrected the training set by weighting existing samples and generating synthetic samples. In contrast to using an LM, they generated synthetic samples using a kernel density estimation (KDE)-based (Silverman 1986), and a SMOTE-based (Chawla et al. 2002) value estimator.

The study of Perez et al. was interesting because they gave insights into generating a diverse set of synthetic questions (Perez et al. 2022). They investigated the use of LMs in red teaming scenarios, where

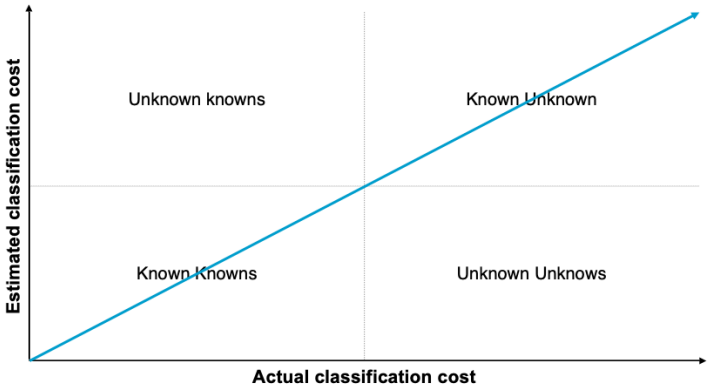


Figure 2.3: An unknown unknown classification error is important to investigate because, in contrast to the estimated classification cost, is expensive. Traditional evaluation methods, such as cross-validated error rates and area-under-the-curve values report the number of known unknown errors (Attenberg, Ipeirotis, and Provost 2015).

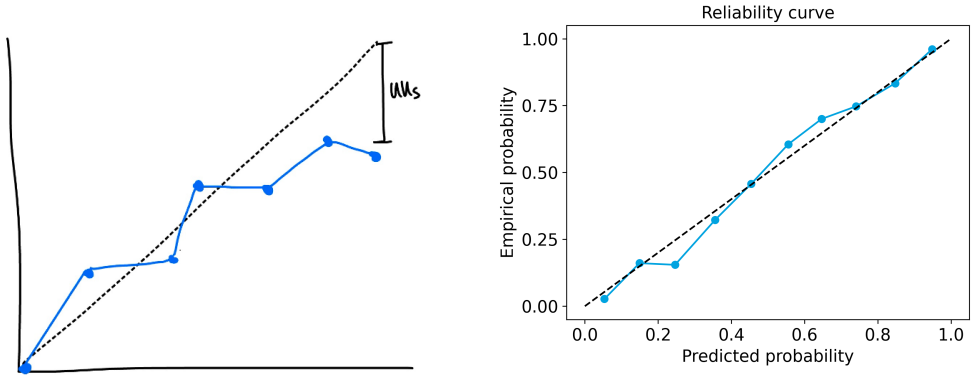
one LM is used to attack another. They experiment with an LM that generates questions to provoke offensive replies from a defending LM. This process involves using a red team classifier to predict the harmfulness of outputs. The study explores various methods, ranging from zero-shot generation to reinforcement learning, to produce test cases of different diversity and difficulty levels. Notably, they focus on generating new examples, aiming to identify instances that are misclassified by the red team classifier. By evaluating the target LM's responses to these generated questions, they uncover numerous offensive replies, highlighting the potential harm of large-scale LM chatbots. Although this study presents interesting insights into generating a diverse set of synthetic questions, it was not incorporated into the methodology or analysis of this study.

2.1.4. Evaluation

Measuring how effectively a system minimizes blind spots is not straightforward. Traditional approaches often overlook blind spots because they focus solely on addressing known unknowns (Attenberg, Ipeirotis, and Provost 2015), in which they look at low confident errors. The literature lacks a consistent standard for measuring the mitigation or coverage of blind spots. Common evaluation techniques might miss certain problems. The question arises as to which metric can accurately represent a reduction in blind spots. The following paragraph discusses how previous studies measured it.

The study of Lakkaraju, which is already discussed, measured the total cost, the number of queries to the oracle and the cumulative regret of the different methods (Lakkaraju et al. 2017). They could measure these metrics, because they grouped the training samples into partitions and measured the entropy between partitions . The lower the with-in entropy and the bigger the between entropy, the better. They also introduced a cost after querying the oracle and were therefore also able to compare the cumulative regret of the different frameworks. Our method refrains from partitioning and does not introduce associated costs, so their metrics cannot adopted. However, we are able to adopt their experiment of OOD data, in which their model was trained on Amazon book reviews and evaluated on Amazon electronics reviews. Instead of Amazon reviews, we used Email and SMS spam detection datasets. Those two datasets not only vary in input length but also differ in class imbalance, increasing the difference even further.

The study of Lippmann et al., who generated synthetic samples, measured the number of UUs occurring after perturbation as a proxy of the number and size of blind spots (Lippmann, Spaan, and Yang 2024). Since we also perturbed and generated input samples, the number of UUs was adopted as a measurement. A downside of this metric occurs when the model's overall confidence drops. This can indicate that the model has no confidence in any of its predictions which is bad, but when only using the number of UUs, it shows an incredible improvement. To combat this, the accuracy and average



(a) Reliability curve before calibration

(b) Reliability curve after calibration

Figure 2.4: An example of a reliability curve. A reliability curve plots the predicted probability on the x-axis, and the empirical probability on the y-axis. In the optimal scenario, when a model is 50% accurate, the model is, on average, 50% correct/incorrect. There is always a deviation between the optimal reliability curve (black dashed line) and the actual reliability curve (blue line). (a) Shows that high-confident predictions are more often incorrect than should be expected based on the model's confidence.

confidence were also measured.

The research conducted by He et al. investigated a method for analyzing blind spots, which involved creating stress tests by generating synthetic data He et al. 2023. They generated a wide range of potential errors and checked whether these samples measured a drop in metric scores. They measured a variety of different measurements and suggested to use a combination of metrics to measure blind spots, as each single metric does not cover everything,. We adopt their recommendation to use multiple metrics and will measure the number of UUs, reliability curve, average confidence, and accuracy. The reliability curve will be explained in the next paragraphs.

One widely used technique for evaluating model calibration is the reliability diagram. The reliability diagram plots the agreement between predicted probabilities and empirical probabilities into one graph (Niculescu-Mizil and Caruana 2005). The resulting line is called the reliability curve. The curve gives insight into the reliability of the model's confidence. In the optimal situation, the reliability curve follows the dashed line in Figure 2.4).The optimal situation indicates that the predicted confidence aligns with the actual accuracy. I.e. when the model's confidence is 80%, it is expected that 80% of those predictions are correct. Figure 2.4a shows a drop in the confidence curve, indicating that the model predicts a too high confidence, i.e., the model predicted a confidence of 80%, but the actual accuracy was 60%. Figure 2.4b show a calibrated confidence that aligns with the optimal curve. The reliability graph does not show the number of UUs, but it does give insight into the proportions in which a confidence prediction is right. Optimizing the reliability curve intuitively decreases the number of UUs, because, one, the number of incorrect high-confident predictions should decrease. Two, the confidence will represent the true prior probability better, i.e., the confidence aligns better with the true performance of the model. There is a research field dedicated to optimizing the reliability curve of predictor models, also called 'calibrating' the reliability curve. This study did not apply those methodologies, as these techniques do not have their main focus on mitigating blind spots and those techniques are always applied as a post processing step. The model itself produces incorrect confidences, which are afterward corrected. The model itself is not performing better. For further reading, the excellent survey of Filho et al. discusses some of the fundamentals of probability calibration and many state-of-the-art methods (Silva Filho et al. 2023). The main focus of this study is to optimize the outputted calibrations of the model itself, we therefore do not dive into probability calibration in too much detail, this could be interesting to investigate in future work. However, in the absence of a uniform metric to measure blind spots, the reliability curve has been calculated.

To measure the calibration of the model, the estimated calibration error (ECE) can be used (Definition 2.1). In the context of the calibration diagram, the ECE provides a single number to show the correctness of the reliability curve. The binary equivalent is calculated following Equation 2.1.4, in which the weighted average over each bin is calculated as $\sum_{n=1}^{N} \frac{|B_i|}{N}$, where B_i is the start and end of the bin and n is the number of samples in the bin. This is multiplied by the difference between the empirical probability $acc(B_i)$ and the predicted probability $conf(B_i)$ (Roelofs et al. 2022).

Definition 2.1: Binary-Estimated Confidence Error

Estimated confidence error ECE is defined as:

$$ECE = \sum_{n=1}^{N} \frac{|B_i|}{n} |acc(B_i) - conf(B_i)|$$

where:

$acc(B_i)$ is the empirical probability,
$conf(B_i)$ is the predicted probability,
$\sum \frac{|B_i|}{n}$ is the weighted average over each bin with n predictions in the bin.

2.2. Adversarial attack

Adversarial attacks refer to deliberate manipulations of input data, designed to deceive predictor models (Morris, Yoo, and Y. Qi 2020). It is used to estimate/improve the robustness of a predictor model. There are multiple ways to attack a predictor model, however, in the scope of this study, it refers to small perturbations of the input samples to cause misclassifications by the LM. After perturbation, the label is still the same as the original unperturbed sample. We only considered a black-box setting, meaning, it does not requires any knowledge about the predictor model or underlying confidences in its prediction. The goal was twofold. First, we counted the number of UUs after perturbation as a robustness metric, and secondly, the discovered UUs were used to generate new synthetic samples. In this section, we describe three adversarial methods, but we only used the black-box adversarial attack method DeepWordBug (Gao et al. 2018).

DeepWordBug was an adversarial attack method proposed by Gao et al. that changed characters in the input text while aiming to maintain the same semantic meaning (Gao et al. 2018). The perturbations were subtle, which stands in contrast to other adversarial methods, like TextFooler, which changed whole words in the input. The method consisted of three key components, namely, scoring the tokenized input, ranking the input, and transforming the tokens. Scoring was done to identify the most important word to perturb, such that the edit distance of the perturbed sentence was minimized. In figure 2.5 an example of a perturbed sentence is shown.

TextFooler inserts and/or replaces words in a sentence (Jin et al. 2020), and was created to evaluate the performance of the BERT model. To decide which word to change they looked at three parts: synonym extraction, POS checking, and Semantic Similarity checking. For synonym extraction, they select the top N number of words that have the most similar word embeddings from Mrksic et al. (Mrkšić et al. 2016). The N candidate words are only kept if they are in the same part of speech POS. The remaining words were each substituted with the highest ranking words forming each an adversarial sample. The similarity of these adversarial samples to the original samples was checked, and only the samples that were more similar to a certain threshold ϵ were kept as final adversarial samples. They evaluated their method on 5 text classification tasks and 2 textual entailment tasks. Including the IMDb dataset.

Unlike DeepWordBug and TextFooler, the research conducted by Dixit et al. (CORE) and Paranjape et al. (RGF) utilized a retrieval algorithm to fetch samples, which were then perturbed. These studies were

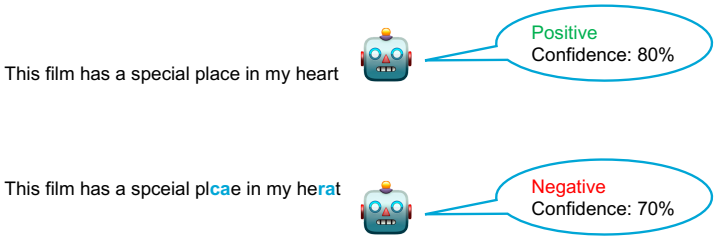


Figure 2.5: Example of an adversarial attack on a input sentence using DeepWordBug. The LM model predicts a different class label with high confidence, while the input sentence has only has a few perturbed characters. Adapted from (Gao et al. 2018; IEEE Symposium on Security and Privacy 2018)

interesting because their method could potentially be incorporated into the generation phase of this study. CORE employed counterfactual data augmentation (CDA) through retrieval and editing mechanisms to introduce a wide array of perturbations (Dixit et al. 2022). They emphasized the effectiveness of CDA in enhancing model generalization on out-of-distribution (OOD) data over other data augmentation (DA) techniques. They integrated retrieval-augmented language models in a two-step process to generate minimally perturbed counterfactuals. First, they retrieved samples and then edited those.

The study of Paranjape et al. used a retrieval algorithm to generate semantically diverse counterfactuals to improve the robustness of LMs for question-answering (QA) tasks (Paranjape, Lamm, and Tenney 2022) . To generate diverse counterfactuals, they proposed the Retrieval, Generate, and Filter (RGF) method. The retrieval step utilizes REALM (Guu et al. 2020), which provides different contexts and answers, which are crucial for creating various alternative scenarios. Like other adversarial methods, RGF generates from question-context-answer (q, c, a) pairs, altered counterparts (q', c', a'). The filtering component ensures noise reduction and preserves question fidelity to the original pair, which is crucial for minimizing synthetic divergence between the two pairs. The central point of Paranjapeet al, is the need for data diversity. Drawing parallels with the present study, Paranjape et al. adopted of REALM as a tool for generating diverse outputs underscoring the transformative potential of retrieval-based methods in augmenting datasets. An interesting take on the retrieval-based approaches could be retrieving alternative samples based on identified UU. This could lead to a more variety of samples and/or possible identification of another blind spot. However, we chose DeepWordBug as an adversarial attack method for its relative simplicity. This allowed us to concentrate more on the additional benefits provided by data valuation, rather than the complexities of the attack method itself. To obtain a diverse set, we generated an explorative hypothesis.

2.3. Data valuation

2.3.1. Definition

Data valuation (DV) in ML is a research area that studies to estimate the 'value' of a (certain) sample or data source. This value can be used to remove or collect data that was found to be of low or high importance for the ML model (Sim, Xu, and Low 2022). It is critical to understand how properties of data influences predictor models, to be able to create more robust models (Hashimoto 2021). Data valuation methods, especially the Shapley value, originate from the cooperative game theory. In cooperative game theory, the question is: "How important is each player to the overall cooperation, and what payoff can he or she reasonably expect?" (Roth 1988). This question has been translated in Computer Science, more specifically machine learning into the question: Which sample contributes what to the final performance of the model? Because ML requires large datasets, this is not a trivial question to answer, especially if you want to answer this in a computationally efficient way. Data values can be used to give insights into which specific sample(s) or data source(s) are important and thus be collected more (Hashimoto 2021). If a sample has a high data value, it means that it's important and a low data value indicates it contributes

less to the performance of the model. In this study, we used the data values to select important samples and perturbed those. Then, the LM was retrained using weighted loss based on data values.

Data valuation strategies are typically divided between two topics. One topic is about determining "fair" compensation for every data owner, such as (Sim, Y. Zhang, et al. 2020). They tried to create a (monetary) incentive to share (more) data or to determine how much the data is worth when you sell/buy data. The other topic considers "interpretable" data valuation strategies, which are mainly concerned with optimizing the model's performance (Ghorbani and Zou 2019; Sim, Xu, and Low 2022). In this study, we only looked at the second strategy: estimating data values for model improvement. This section is divided into 4 subsections, each describing a type of data valuation technique: leave-one-out, cooperative game theory, meta-learning, and reputation-based methods.

2.3.2. Leave-one-out

One of the most basic methods, and often used as a baseline, is the Leave-one-out method (Ghorbani and Zou 2019). The method originates from 1977 and is easy to understand. It compares the model's performance trained on all samples in a dataset, to the model's performance when leaving one sample out (Cook 1977). The difference in performance, also called marginal contribution, is the data value of that particular sample. It is simple, however, it is understandable that this does not scale well to large datasets. It would imply retraining an LM with and without each sample. Besides the computational cost, it is debatable if the performance change of leaving out one sample on an LM can be measured. Therefore, LOO is not suitable for a scalable solution and is not used in our study.

2.3.3. Cooperative game theory

In cooperative game theory, players can form coalitions and work together to achieve a common goal, which is maximizing the total reward for the group. The theory focuses on how to distribute the collective reward amongst the players of the coalition fairly, based on individual contributions. Cooperative game theory can be adapted to calculate the value, or contribution, of individual samples to the model performance. We will first discuss the most used strategy in cooperative game setting, the Shapley value. Then, we will discuss studies that adapted the Shapley value and applied it to machine learning.

In cooperative game theory, some players can play together and form coalitions. Each coalition can have a different reward. Like LOO, the Shapley value calculates the marginal contribution and does this by using the weighted average of a player's contributions in all the coalitions that a player can play (Lundberg and Lee 2017). The Shapley value (Definition 2.2), consists of two components: marginal contribution (Equation 2.5) and weight (Equation 2.6). The marginal contribution calculates the contribution of a certain player i in each coalition S it can join. The weights come from the probabilities for the respective marginal contribution, which is simply the number of ways the same coalitions can be formed. E.g., a coalition consisting of player 1 and player 2 can be formed in 2! ways: $c_{1,2}$ and $c_{2,1}$.

An example of a cooperative game is shown in Table 2.1. It shows that when no players form a coalition c_{none}, the coalition earns no reward. When the players play solo, the reward for player 1 is $7,500$, and for player 2 it's $5,000$. If both players collaborate in the same coalition c_{12}, the total reward is $10,000$. Although the total reward increased when both players formed a coalition together compared to playing solo, it intuitively makes sense that player 1 contributed more to the joined coalition. This is because player 2 gained more rewards when collaborating. Following Definition 2.2, we can calculate the Shapley values for players 1 and 2.

For player 1 we will get:

- $c_{1,2} - c_2 = 5,000$
- $c_1 - c_{empty} = 7,500$

The marginal contribution (Shapley value) for player 1 is $(5,000 + 7,500)/2 = 6,250$

For player 2 we get:

- $c_{1,2} - c_1 = 2,500$
- $c_1 - c_{empty} = 5,000$

The marginal contribution (Shapley value) for player 2 is $(2,500 + 5,000)/2 = 3,750$

You might have observed that the sum of marginal contributions adds up to the maximum attainable reward of $10,000$. This is always true, as the objective of a cooperative game is to maximize the overall reward for the group, rather than focusing on individual player rewards. When applying the Shapley value to a machine learning problem, the reward is represented by the best performance of a machine learning model, and the individual players correspond to individual samples or individual data features.

Table 2.1: A cooperative game, in which players 1 and 2 took part. The reward is the highest when both players collaborate. Intuitively it makes sense that player 1 contributes more to the total reward when taking part in the joined coalition, as player 2 gained more when collaborating.

Coalition	Reward
$c_{1,2}$	10,000
c_1	7,500
c_2	5,000
c_{none}	0

2.2: Shapley value

For a player i in a cooperative game with a set of players N, the Shapley value for player i, denoted as ϕ_i, is defined by the following equation:

$$\phi_i = \sum_{S \subseteq N\{i\}} \frac{|S|!\,(|N| - |S| - 1)!}{|N|!} \left[\text{val}(S \cup \{i\}) - \text{val}(S)\right] \tag{2.4}$$

where Equation 2.5 is the marginal contribution after adding player i to coalition S. Here, val(S) is the value of coalition S, and val$(S \cup \{i\})$ is the value of coalition S including player i.

$$\left[\text{val}(S \cup \{i\}) - \text{val}(S)\right] \tag{2.5}$$

Equation 2.6 is a weighting factor that accounts for the number of permutations of players in which subset S can occur before player i joins. This ensures that each subset's contribution is appropriately scaled and the overall formula is normalized considering all possible orders of player entry.

$$\frac{|S|!\,(|N| - |S| - 1)!}{|N|!} \tag{2.6}$$

Here, $|S|$ denotes the number of players in coalition S, $|N|$ is the total number of players, $|S|!$ represents the number of ways coalition S can be formed, and $(|N| - |S| - 1)!$ is the number of ways players can join after player i.
The contribution is calculated for each possible subset $S \subseteq N \ \{i\}$, indicated by:

$$\sum_{S \subseteq \{1,\ldots,N\}\{i\}}$$

Definition created based on (Lundberg and Lee 2017)

There is a conflict between studies saying the Shapley value (SV) is too computationally complex and cannot be used in practical applications (Yoon, Arik, and Pfister 2020) and studies saying that the SV is practical and usable (Ghorbani and Zou 2019; Li et al. 2023). The naive implementation of the

Shapley value grows computationally exponential as the number of participants increases because it involves calculating the expected reward for every possible subset of players. This complexity makes it unfeasible to compute the Shapley value for scenarios involving a large number of participants(Li et al. 2023). However, other studies estimate the Shapley value.

The study of Yan et al. proposed to use the lesser-known "Least core" approach (Yan and Procaccia 2021). The Least Core is a linear program and is, similar to the SV, computationally expensive to calculate. However, it can be relaxed in several ways to improve efficiency. Another alternative for the SV is the Banzhaf value (D. Han et al. 2023; Sim, Xu, and Low 2022), which is defined as the average marginal contribution of a participant compared to all other participants. It is different compared to the Shapley value in that it does not possess the game-theory characterization of "Efficiency" (D. Han et al. 2023), but only possesses the property of "2-Efficiency" (Lehrer 1988). Efficiency means that the sum of the values of all participants is not equal to the sum of values if participants train the model separately (D. Han et al. 2023). 2-Efficiency means that if two participants are merged into one, their values do add up (D. Han et al. 2023). The Banzhaf value is less popular. At the start of this study, we compared the speed of the Data Shapley method to the speed of Data valuation using Reinforcement Learning (DVRL) method. More about the DVRL method in the meta-learning section, section 2.3.5

2.3.4. Reputation-based methods

Reputation-based methods consider the reliability and credibility of data sources when assessing the value of data. These methods typically involve assigning a reputation score to each data provider, The methods are mostly built to rank or value a data source, instead of a single sample, however, some may be adjusted to do so. Originally used by Google to rank web pages in search results, PageRank (Page et al. 1999) can be seen as a reputation-based method. It values web pages based on the number and quality of links to them, essentially using the reputation of linking pages to infer the value of a page. Here, a single page can be seen as a sample, instead of a data source.

Reputation can be used to create a weighted average of the data sources and/or affect which samples are selected in the next training round (Shi, Yu, and Leung 2023). The studies using reputation-based (Kang, Xiong, Niyato, Xie, et al. 2019) and endorsement-based (Kang, Xiong, Niyato, Yu, et al. 2019) data valuation apply this second approach. In (M. Qi et al. 2022) reputation is used to select data sources with a higher reputation with more probability. The study of (J. Zhang, Y. Wu, and Pan 2021) used auctions to determine which data source to select. They value participants with high-quality data and many contributions with high reputations. Although reputation-based methods are mostly used to determine which data source should be selected, they can be used to sort individual samples.

2.3.5. Meta-learning

Meta-learning-based algorithms are algorithms that learn from past experiences (Vilalta and Drissi 2002). Meta-learning algorithms for data valuation, such as DAVIZ (Z. Wu, Shu, and Low 2022) or Data Valuation using Reinforcement Learning (DVRL) (Yoon, Arik, and Pfister 2020), estimate the data values and the training procedure into one algorithm. To estimate the data values, the study of Yoon et al. uses a reinforcement learning algorithm to learn from the reward coming from performance improvements on a small validation set (Yoon, Arik, and Pfister 2020). The DVRL method combines the data valuation and model training steps but uses the MAML (Finn, Abbeel, and Levine 2017) framework in their algorithm. Finn et al. then tried to optimize the data valuation, such that the outcome of the model gains the best performance. They learn from historical data and feature sharing and achieve state-of-the-art results.

In this study we use the data valuation method DVRL and, therefore, we discuss the method in more detail. The DVRL method consists of two learnable functions, the predictor model f_θ, and the Data Value Estimator (DVE) model h_ϕ. The method works in iterations, where the DVE model h_ϕ selects a batch of samples that will be shown to the predictor model f_θ for that iteration. To determine which samples should be shown, h_ϕ learns a selection probability $w \in [0, 1]$ for each sample in the training set.

$$f_\theta : X \to Y$$
$$h_\phi : X \times Y \to W$$
$$W \in [0, 1]$$

The predictor model f_θ can be any predictive model that can take an input X and output Y. The predictor model is updated by minimizing some loss function $\mathcal{L}_f$. Depending on the chosen model, usually a Deep Neural Network (DNN), optimization can be done using standard gradient descent. The DVE model h_ϕ takes as input X and Y and outputs a weight W. The loss of the DVE model $\mathcal{L}_h$ cannot be optimized using standard gradient descent. This is because the model selects a subset of samples, which is represented by a boolean 1 or 0 and is not differentiable. Instead, the DVE model is updated using a reinforcement signal. This signal is based on the moving average loss of the predictor model on a validation set according to Equation 2.7. The optimization tries to minimize the expected loss on the validation set V, where the validation set has the same distribution as the target set $V \sim P^{target}$. It minimizes the loss when it selects samples for the predictor model, from which the predictor model decreases the loss on the validation set. This implies that the DVE model is incentivized to select samples that look like the validation set distribution.

$$\begin{aligned} &\min_{h_\phi} \mathop{\mathbb{E}}_{(\mathbf{x}^v, y^v) \sim P^t} \left[\mathcal{L}_h \left(f_\theta \left(\mathbf{x}^v\right), y^v\right)\right] \\ \text{s.t. } f_\theta = &\arg\min_{f \in \mathcal{F}} \mathop{\mathbb{E}}_{(\mathbf{x}, y) \sim P} \left[h_\phi(\mathbf{x}, y) \mathcal{L}_f(\hat{f}(\mathbf{x}), y)\right] \end{aligned} \tag{2.7}$$

where

$$x^v, y^v \in V$$

Because the loss is nondifferentiable, Yoon et al. introduced a reinforcement signal to optimize the DVE model. The agent is the DVE model that selects a batch of samples. The reward is the evaluation of the predictor model's evaluation on the validation set after it has learned from the batch of samples selected in that iteration. The loss of the DVE model in a single iteration is calculated following Equation 2.8. Here, s are the selected samples in that iteration, and $\pi_\phi(D, s)$ is the probability that the selection vector s is selected, based on the selection probabilities W of h_ϕ itself. The gradients are calculated following the equation:

$$\begin{aligned} \hat{l}(\phi) &= \mathop{\mathbb{E}}_{(\mathbf{x}^v, y^v) \sim P^t, \mathbf{s} \sim \pi_\phi(\mathcal{D}, \cdot)} \left[\mathcal{L}_h \left(f_\theta \left(\mathbf{x}^v\right), y^v\right)\right] \\ &= \int P^t \left(\mathbf{x}^v\right) \sum_{\mathbf{s} \in [0,1]^N} \pi_\phi(\mathcal{D}, \mathbf{s}) \cdot \left[\mathcal{L}_h \left(f_\theta \left(\mathbf{x}^v\right), y^v\right)\right] d\mathbf{x}^v \end{aligned} \tag{2.8}$$

where

s are the selected samples for that iteration, which is binary encoded. $s \in \{0, 1\}$. $\pi_\phi(D, s)$ is the probability that the selection vector s is selected based on $h_\phi(D)$

which has the gradient:

$$\begin{aligned} &\nabla_\phi \hat{l}(\phi) = \mathop{\mathbb{E}}_{\substack{(\mathbf{x}^v, y^v) \sim P^t \\ \mathbf{s} \sim \pi_\phi(\mathcal{D}, \cdot)}} \left[\mathcal{L}_h \left(f_\theta \left(\mathbf{x}^v\right), y^v\right)\right] \nabla_\phi \log\left(\pi_\phi(\mathcal{D}, \mathbf{s})\right), \\ &\text{where } \nabla_\phi \log\left(\pi_\phi(\mathcal{D}, \mathbf{s})\right) \\ &= \nabla_\phi \sum_{i=1}^{N} \log\left[h_\phi\left(\mathbf{x}_i, y_i\right)^{s_i} \cdot \left(1 - h_\phi\left(\mathbf{x}_i, y_i\right)\right)^{1 - s_i}\right] \end{aligned}$$

The gradient derivation is provided in the paper by Yoon et al. (Yoon, Arik, and Pfister 2020). The selection probabilities W, generated by the Data Value Estimator (DVE) model h_ϕ represent the final data values. For a single sample x_i, it can be calculated as $w_i = h_\phi(x_i, y_i)$.

The computational complexity of the DVRL is determined by the number of iterations and the duration of each iteration. Initially, we aimed to use the pre-trained BERT language model as the predictor model. However, this would require retraining the model in every iteration, which is impractical. Consequently, we chose to deviate from the original implementation and employ a simpler predictor model that takes BERT embeddings as input. This allows us to utilize the contextual representation of the BERT model without the need to retrain the entire model in each iteration. The drawback of this choice is that we do not utilize the predictor model that has been jointly optimized with the DVE model. In the next chapter, we delve into the implementation details.

Method

As concluded in the chapter 2, there is a need for scalable methods to mitigate blind spots when training a model. Lippmann et al.'s approach, which involves generating synthetic samples based on UUs, demonstrates the potential to reduce these blind spots. Their strategy perturbs the validation set to elicit UUs by an LM and then selects a random subset of those UUs. This chapter will delve into the proposed approach that aims to enhance the process of selecting the most suitable UUs. The literature suggests that OOD data is one of the causes of blind spots. Therefore, we propose to use data valuation (DV), a research field that assesses the importance of each sample to the model's performance. DV can identify biased, noisy, or OOD samples in the training set (Sim, Xu, and Low 2022). In our methodology, the DV method requires a validation set that represents a target distribution. In addition to optimizing the selection procedure, a second step of the model retrained the LM using a Weighted Loss based on Data Values (WLDV). This chapter discusses the method, starting with a definition of blind spots in section 3.1. Section 3.2 explains the four datasets and tasks used to conduct the experiments. Section 3.3 shows how data values are estimated for each sample. Then, section 3.4 describes how adversarial attack has been applied to elicit UUs. Section 3.5 explains the generation of synthetic samples. Selection 3.6 explains the introduction of data values when retraining the model after the dataset has been extended with the synthetic samples. Finally, section 3.7 discusses the four evaluation combinations, such that we can compare the contribution of each step of the method.

In figure 3.1 our method is depicted. It starts with a training set and values each sample using DV. The highest valued samples are perturbed using adversarial attack. The resulting UUs are used to generate synthetic samples according to Lippmann's method (Lippmann, Spaan, and Yang 2024) and are used to extend the original training set. Then the second part of the methodology calculates the data values again on the extended training set. These values are then used to retrain the LM using weighted loss based on data values (WLDV). The following sections describe the steps in more detail.

3.1. Blind spot: definition

Before discussing the blind spot mitigation technique, its necessary to formalize the definition, as there is some ambiguity in the literature. This study defined blind spots as a cluster of unknown unknowns (UUs), which is in line with other studies (Lippmann, Spaan, and Yang 2024; Lakkaraju et al. 2017; Bansal and Weld 2018). Some use the term blind spot and UUs interchangeably (Cabrera et al. 2021), while others use the term 'unknown unknown' to indicate a test sample that is unseen during training time (Chung et al. 2019). The definition of a blind spot is not precise, but it is intended to convey a more informal understanding of the relation between UUs and blind spots.

We maintain a more specific definition for UUs in the context of a predictor model. Let x be a sample in the problem space D. Each sample x has a "true" label y from a set of Y. Without losing generalizaiblity, we use a classification problem, which means $Y = \{0, 1\}$. The task of the predictor model θ is to estimate a label $\hat{y}$, such that it belongs as closely to the true label y. In this study we consider the predictor model θ to output posterior probabilities for both classes, i.e., $p(1|x)$ and $p(0|x)$. The posterior probabilities always sum to one $p(1|x) + p(0|x) = 1$. The highest posterior probability represents the predicted class, following Equation 3.1.

$$\hat{y} = \begin{cases} 1 & \text{if } p(1|x) > p(0|x) \\ 0 & \text{otherwise} \end{cases} \tag{3.1}$$

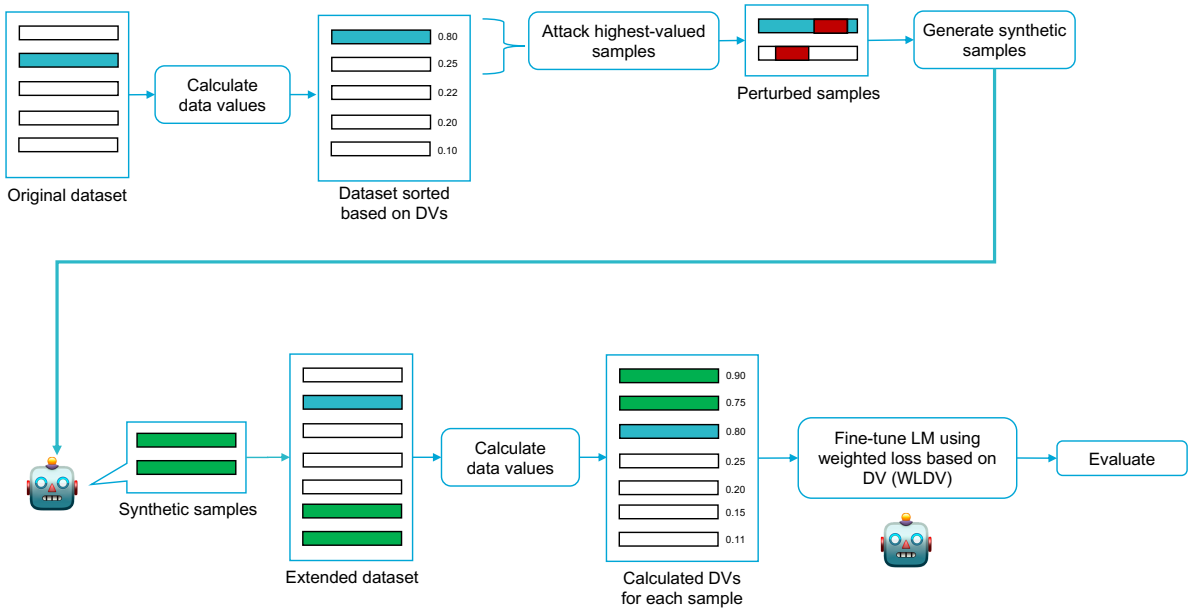


Figure 3.1: The proposed method consists of multiple steps. First, the data value of each sample in the training is calculated using the DVRL method. Then, the highest-valued samples are attacked using DeepWordBug. Based on the resulting UUs, synthetic samples are generated. The original training set is extended using those synthetic samples and the data values are re-calculated for each sample in the extended training set. Finally, the LM is retrained using a weighted loss based on the re-calculated data values.

The highest posterior probability represents the confidence c the predictor model has in its prediction, shown in Equation 3.2. In line with previous works (Lakkaraju et al. 2017; Bansal and Weld 2018; Lippmann, Spaan, and Yang 2024), we use the threshold $\tau \geq 0.65$ as a high confidence classification. A UU is now defined as a prediction where the predicted class is not the true class, $\hat{y} \neq y$, and the confidence c is higher than τ. The misclassification cost for both errors is the same. The set of unknown unknowns U is defined following Definition 3.1.

$$c = \begin{cases} p(1|x) & \text{if } p(1|x) > p(0|x) \\ p(0|x) & \text{otherwise} \end{cases} \tag{3.2}$$

3.1: A set of Unknown Unknowns

Given a set of predictions $P = \{(x_i, \hat{y}_i, c_i)\}_{i=1}^{n}$ made by a model θ, the set of unknown unknowns U is defined as:

$$U = \{(x_i, \hat{y}_i, c_i) \in P \,|\, \hat{y}_i \neq y_i \text{ and } c_i \geq \tau\} \tag{3.3}$$

where

$D = \{(x_i, y_i)\}_{i=1}^{n}$ (original dataset)
$x_i =$ input sample i
$y_i =$ corresponding true label of x_i
$\hat{y}_i =$ predicted label for x_i
$c_i \in [0, 1]$ (confidence score predicted by model θ)
$\tau = 0.65$ (confidence threshold)

Table 3.1: Four datasets have been used for the experiments. The IMDb, MRPC, QNLI, and the combined Email-SMS dataset. The QNLI dataset is the largest dataset of them all. The validation set will be used to calculate the data values.

Name	Training	Validation	Synthetic	Test	Task
IMDb	25,000	500	500	25,000	Sentiment Analysis
MRPC	3,670	408	63	1,730	Paraphrase Identification
QNLI	116,000	5,460	1160	5,460	Question Answering
Email-SMS	31,716	317	625	5574	Spam Detection

3.2. Datasets

Four datasets have been used to show that the method works robustly under a variety of tasks: IMDb for sentiment analysis (Maas et al. 2011); MRPC for paraphrase identification (Dolan and Brockett 2005); QNLI for question-answering (Rajpurkar et al. 2016); and Email-SMS to simulate domain adaptation (Wiechmann 2023; Almeida and Hidalgo 2012). Table 3.1 shows the composition of the different datasets, showing the train/validation/test split, the number of synthetically generated samples, and the dataset task. The validation set should be seen as data that is generated in the deployment phase. A validation was created if a dataset did not have a pre-defined validation set.

The IMDb dataset[1], derived from the Internet Movie Database, is often used for sentiment analysis in machine learning research. Comprising movie reviews, it offers a diverse range of opinions and emotions, making it valuable for training and evaluating LMs on tasks related to understanding sentiment in textual data (Maas et al. 2011). A validation set was created using a random subset of 500 samples selected from the test set. The test set was chosen because the data should represent the data during the deployment phase. It is not directly used to retrain the LM, only to calculate the data values.

The MPRC dataset[2] stands for the Microsoft Research Paraphrase Corpus (Dolan and Brockett 2005) and is focused on paraphrase identification. MRPC assesses the model's ability to distinguish the similarity between sentences. This dataset is crucial for understanding how LMs capture nuanced variations in meaning and expression.

The QNLI dataset[3] stands for Question-answering Natural Language Inference (Rajpurkar et al. 2016). Centered around natural language inference tasks, QNLI involves pairing questions with sentences, providing a platform to evaluate a model's comprehension of contextual relationships. This dataset contributes to the exploration of LMs in the context of understanding and answering questions, reflecting their proficiency in handling diverse linguistic challenges.

The Email-SMS datasets are two independent datasets, which have the same task: spam detection. The LM was trained on the Email dataset (Wiechmann 2023) and evaluated on the SMS dataset (Almeida and Hidalgo 2012). The goal was to have two related datasets to simulate out-of-distribution data. The validation set was created by taking 317 samples from the test set. The training set consisted of 31716 samples and the test set of 5574 samples. For the email dataset, the subject and message were concatenated by the [SEP] symbol. We chose the Email and SMS datasets because these two datasets deviated from each other but have the same task. The SMS test set was unbalanced, with 86% of the samples being spam. The Email training set was almost balanced, 16163 spam (51%) and 15553 not spam (49%). The email dataset consists of long messages and the SMS consists of short sentences. This dataset was included, because other studies also performed domain adaptation (Lakkaraju et al. 2017).

[1]download link IMDb: https://huggingface.co/datasets/imdb
[2]download link MRPC: https://huggingface.co/datasets/glue
[3]download link QNLI: https://huggingface.co/datasets/glue

3.3. Data valuation

The data values had been estimated twice: once to calculate the data values before synthetic samples were added to the training set; and once after the synthetic samples were added to the training set. We extended the method Data Valuation using Reinforcement Learning (DVRL) (Yoon, Arik, and Pfister 2020). The details of the DVRL method are discussed in the related work section 2.3.5, in this section, we discuss how we extended and deviated from their implementation.

The computational complexity of the DVRL method was determined by the number of iterations and the duration of each iteration. The results of the DVRL study indicated that more iterations seemed to show a better performance. Following related work (Jiang et al. 2023), we choose to use 2,000 iterations. The iteration would become too computationally complex if the pre-trained BERT language model were used as a predictor model. This is due to the necessity of retraining the BERT model every iteration. Consequently, we used an alternative approach. Instead of using the predictor model as the final model, we used the prediction model as an intermediate model. The model was only used to calculate the data values and then disregarded. We selected a simple logistic regression model as a predictor model. This model might not predict as well as the BERT model, but it allows us to optimize it fast. To still utilize the BERT contextual information, the model took BERT embeddings as input. The BERT embeddings only needed to be calculated once. This allowed us to utilize the contextual representation of the BERT model without needing to retrain the entire model in each iteration. The drawback of this choice was that we did not utilize the predictor model that has been jointly optimized with the DVE model.

We have chosen the logistic regression model as a predictor model because of its simplicity and computational efficiency. By using the pre-trained BERT embeddings as inputs, we still kept the encoded representations from the LM. The embeddings were taken by selecting the last activation layers of the BERT model.

The DVE model h_ϕ could, like the prediction model, be any learnable model. We implemented it in two parts. The first part was a multi-layer perceptron mlp, consisting of 10 layers that alternated between a linear layer and the rectified linear activation unit (ReLU). The mlp had as input a training sample x_i and accompanying label y_i. The second part of the model combined the predicted label $\hat{y}$ of the predictor model f_θ, with the output of the mlp with the predicted label $\hat{y}$.

The intermediate predictor model was not used in future steps. For the final predictor model, we used the pre-trained BERT model. To still use the data values, we opted to fine-tune the pre-trained BERT LM once, after the calculation of the DV had been calculated. To do this, we introduced a Weighted Loss based on Data Values (WLDV). A more elaborate discussion how the LM was retrained can be found in section 3.6. Our approach is shown in Figure 3.2. The iteration is a simplified version of the original figure in the DVRL paper, but now it is extended with a separate predictor model. For IMDb, the *bert-base-uncased* model was used, but for the other datasets, the *distilbert-base-uncased* model was used as the predictor model. This was done to speed up calculation time.

The basis of the DVRL was implemented using the framework of Jiang et. al. (Jiang et al. 2023). They have implemented multiple data valuation strategies, which makes it possible to switch and compare between different data valuation methods. Some imperfections were found, which are mentioned in the limitation and recommendation chapter 6. It includes stochasticity in the resulting data values, which were not expected, as it is common to have the same data value for each sample (Sim, Xu, and Low 2022).

3.4. Adversarial attack

After calculating the data values for each training sample, the highest-valued samples were perturbed using the adversarial attack method DeepWordBug (Gao et al. 2018). The hypothesis was that those UUs would contribute more to mitigating blind spots compared to randomly selecting UUs. The aim is

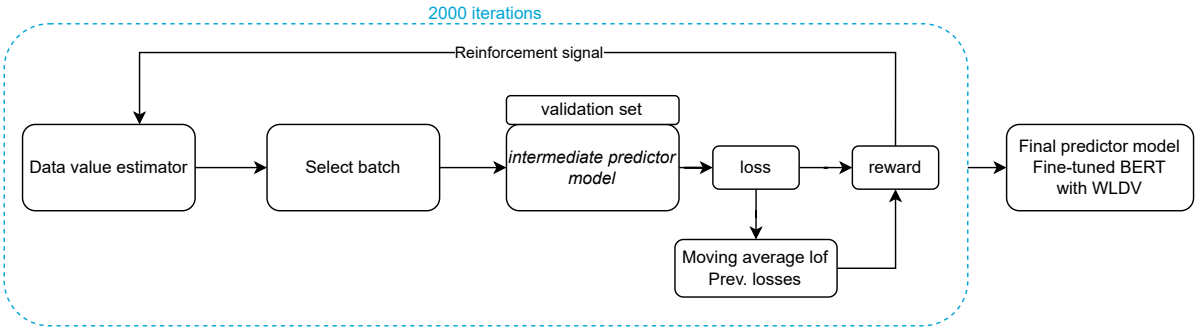


Figure 3.2: The data values are calculated in 2,000 iterations. In each iteration, the learnable Data Value Estimator (DVE) h_ϕ model selects a batch of samples. Those samples are used to train the predictor model f_θ. The loss of the DVE model is linked to the performance of the predictor model on a validation set. After the DVE has been trained and estimated the data values of each sample, the LM is retrained once using weighted loss based on data values. The figure is based on (Yoon, Arik, and Pfister 2020).

to find UUs in a proportion equivalent to 1% of the training set size. DeepWordBug is a state-of-the-art perturbation method that generates small character-level perturbations Δx in a black-box setting, intending to perturb the input sentence x such that the LM model misclassifies the input text. A successful perturbed sample is defined according to definition 3.2.

Gao et al. proposed a scoring mechanism to identify the critical token that, if modified, causes a misclassification by a model, keeping into account to minimize the edit distance between the original sentence x and the perturbed token x'. The scoring method assumed an input sequence $x = x_1x_2...x_n$, where x_i represents the token at the i^{th} position. The score of token x' was determined by combining two methods: the Temporal Head Score (THS) and the Temporal Tail Score (TTS). Both scores were determined by the difference between the model predictions with and without a token. The THT score added one token at a time and compared it to leaving out the token, shown in equation 3.4. This determined the score for the added token x_i. The same was done by calculating the TTS score, only now adding the token starting from the end. So, comparing the predictions after prepending the token of interest and comparing it to leaving that token out, shown in equation 3.5. The combined score, see equation 3.6, was the score used by DeepWordBug. It combined both the THS and TTS, where λ was a hyperparameter.

$$\text{THS}\left(x_i\right) = F(x_1, x_2, \ldots, x_{i-1}, x_i) - F(x_1, x_2, \ldots, x_{i-1}) \tag{3.4}$$

$$\text{TTS}\left(x_i\right) = F(x_i, x_{i+1}, x_{i+2}, \ldots, x_n) - F(x_{i+1}, x_{i+2}, \ldots, x_n) \tag{3.5}$$

$$\text{CS}\left(x_i\right) = \text{THS}\left(x_i\right) + \lambda\left(\text{TTS}\left(x_i\right)\right) \tag{3.6}$$

The DeepWordBug method employed four transformations to perturb a sentence: swapping adjacent letters, substituting a character with a random character, deleting a random character, and inserting a random character. It imposed constraints to avoid repetitive changes in words or alterations to stopwords, with a maximum edit distance set to 30. The alternations were through a greedy word swap search guided by word importance. If these failed, it attempted to swap a random character within the set edit distance. The method outputted the original and perturbed results, along with their corresponding scores,

before and after perturbation, respectively, measured by the $UntargetedClassification$, which aimed to minimize the score of the correct label until a misclassification occurred.

Definition 3.2: Successful perturbation

Suppose a classifier $F(\cdot) : \mathbb{X} \rightarrow \mathbb{Y}$ original sample is x, an adversarial example x' in Untarget attack follows:

$$x' = x + \Delta x || \Delta x ||_p < \epsilon, x' \in X$$
$$F(x) \neq F(x')$$

where

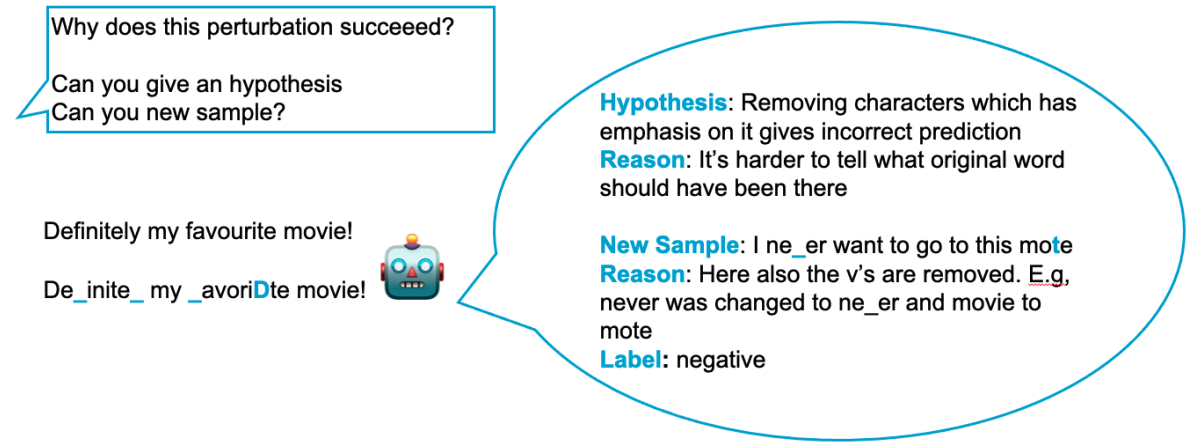
$$\epsilon = \text{maximum edit distance}$$
$$\Delta x = \text{Edit distance}(x, x')$$

3.5. Synthetic sample generation

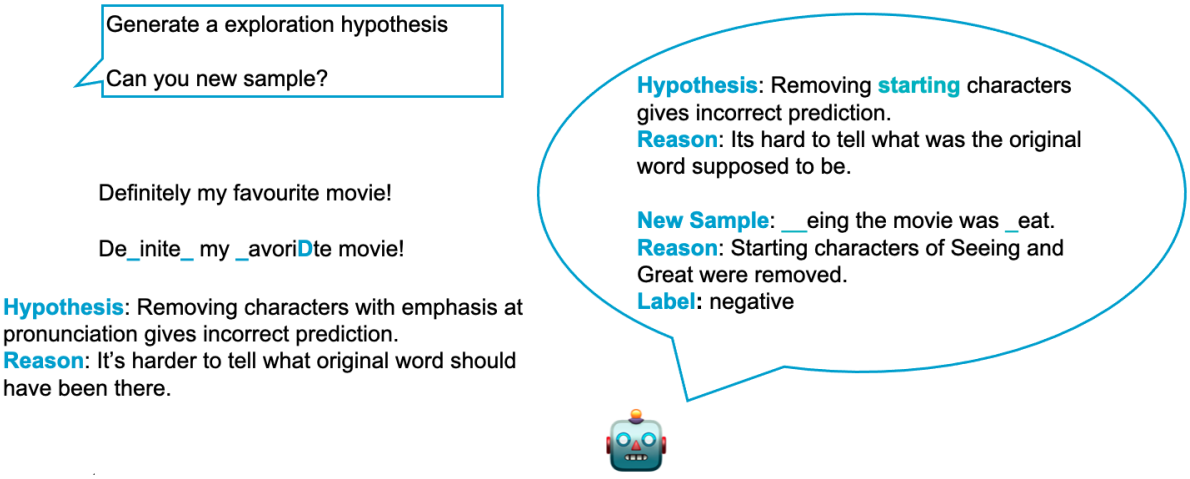
We leveraged the UUs found using the method discussed in the previous section, to generate synthetic samples. Lippmann et al. showed that LMs could generate synthetic samples that reduced blind spots (Lippmann, Spaan, and Yang 2024). Similar to their work, we used OpenAI's *text-davinci-003* to formulate a "generative hypothesis" giving it the original unperturbed sample and the perturbed sample. The generative hypothesis aimed to describe the blind spot in which the UUs sample was found. To find additional blind spots, an "exploration hypothesis" was formulated using OpenAI's *text-davinci-003*, giving it the original sample, the perturbed sample, and the generative hypothesis. Using both hypotheses, OpenAI's *text-davinci-003* was asked to generate two new synthetic samples. In total 2% of synthetic samples were generated from the training set. The training set was extended with the synthetic samples and then shuffled to prevent the LM from learning the sequence of the dataset. The specific prompts can be found in the appendix A.

In line with previous work (Lippmann, Spaan, and Yang 2024), we opted to select the number of UUs to be discovered as 1% of the training set size. Because not all the perturbed samples would become UUs, this in practice led to attacking around 5% of the training set on average. To evaluate the model's performance on the DeepWordBug method, a fine-tuned model was used. Otherwise, the model would have too little confidence in any of their predictions, causing no UUs. The models were fine-tuned using the same settings as the ModelZoo of Textattack[4], but fine-tuned for 2 epochs, instead of 5. Figure 3.3aa shows the "generative hypothesis" generation step. Figure 3.3ab shows the "exploration hypothesis" step. The figures do not contain the exact prompts, these can be found in the appendix **??**.

[4]ModelZoo URL https://textattack.readthedocs.io/en/latest/3recipes/models.html



(a) The hypothesis generation step.



(b) The hypothesis exploration step.

Figure 3.3: (a) OpenAI's *text-davinci-003* was asked to generate a hypothesis and a synthetic sample based on a UU example found after perturbing the training set. (b) OpenAI's *text-davinci-003* was asked to generate an exploration hypothesis and a synthetic sample using the generation hypothesis generated in the previous step. The synthetic samples generated in both steps were used to extend the training set. *The questions in the figures are not the actual prompt send to OpenAI. For the exact prompts, see appendix A*

3.6. Fine-tune Language model

Originally, the LM would be fine-tuned together with the data value estimation according to equation 3.7, where f_θ is the predictor model and $h_\phi(x_i, y_i)$ is the output of the data value estimator (DVE) for sample i (Yoon, Arik, and Pfister 2020). The $\mathcal{L}_f\left(\hat{f}\left(\mathbf{x}_i\right), y_i\right)$ is the loss of the predictor model itself. Since the LM is fine-tuned after the data values have been calculates, this means $\mathcal{L}_f$ is the loss during fine-tuning. The commonly used cross entropy loss was used as loss function. The DVE model outputs data values $h_\phi : X \times Y \to [0, 1]$.

$$f_\theta = \arg\min_{\hat{f}\in\mathcal{F}} \frac{1}{N}\sum_{i=1}^{N} h_\phi\left(\mathbf{x}_i, y_i\right) \cdot \mathcal{L}_f\left(\hat{f}\left(\mathbf{x}_i\right), y_i\right) \tag{3.7}$$

The DVRL method assumed two learnable functions, f_θ and h_ϕ, but we already calculated the data values. We therefore propose equation 3.8 , instead of 3.7. The trainable function h_ϕ has been changed for a constant value dv_i, which resembles the calculated data value for sample i that has been calculated in the previous step.

$$f_\theta = \arg\min_{\hat{f} \in \mathcal{F}} \frac{1}{N} \sum_{i=1}^{N} dv_i \cdot \mathcal{L}_f \left(\hat{f}\left(\mathbf{x}_i\right), y_i \right) \tag{3.8}$$

The parameters used during retraining correspond to The fine-tuning process used training arguments copied from corresponding Textattack (Morris, Yoo, and Y. Qi 2020) settings, with the only modification being a reduction in the number of epochs from 5 to 2. These Textattack parameters have demonstrated effective in the same tasks and datasets. The model was trained on a single RTX2080 GPU, and accuracy as well as the number of UUs were calculated as metrics. The cross-entropy loss function, an industry standard for classification problems, was used. With the exception of IMDb, all models utilized the *distilbert-base-uncased* (Sanh et al. 2020) model from Huggingface; for IMDb, *bert-base-uncased* (Devlin et al. 2019) was used. To ensure code reproducibility, the settings are available in the repository in a designated file [5]

Sorting the training data from high-valued to low-valued samples during batch training may be advantageous. This approach aims to address potential issues arising from outliers in the training data, as clustering high- and low-valued samples together can help mitigate this problem when calculating the average loss per batch.

3.7. Evaluation

To determine if our proposed method reduced blind spots and outperformed other methods, we compared the performance different combinations. To recap, our method consisted of two steps. The first step (section 3.5) extended the training set by extending it using synthetic generated samples based on high-valued training samples. We reference this step as BLISS. We compared the synthetic generation step with the approach of Lippmann et al., who generated the sampled based on randomly selected UUs. We refer their method as IBS. The second step in our approach retrained the model using weighed loss based on data valuation (section 3.6). We refer to this step as WLDV.

Figure 3.4 shows the different combinations that are compared to each other. It shows which steps are taken before the predictor model is evaluated. We will now discuss each sub figure. In Figure 3.4a we show the different combinations we have evaluated. Figure 3.4a in the situation in which the training set is not extended and the LM is fine-tuned without weights. In Figure 3.4b the training set is not extended, but the LM fine-tuned using WLDV. Figures 3.4c and 3.4e the training set are extended using IBS and BLISS respectively. In both situations the LM is not fine-tuned with WLDV. In Figures 3.4d and 3.4f, the training set is extended using IBS and BLISS respectively, and the LM is fine-tuned using WLDV. Five key metrics were measured: the accuracy before perturbation, the average confidence, the accuracy after perturbation, the number of unknown unknowns (UUs) and the reliability curve. The results can be fond in the next chapter.

[5] https://github.com/heblol/tudelft-cs-thesis-wis/tree/main/notebooks/experiment_7_put_together/train_settings.py

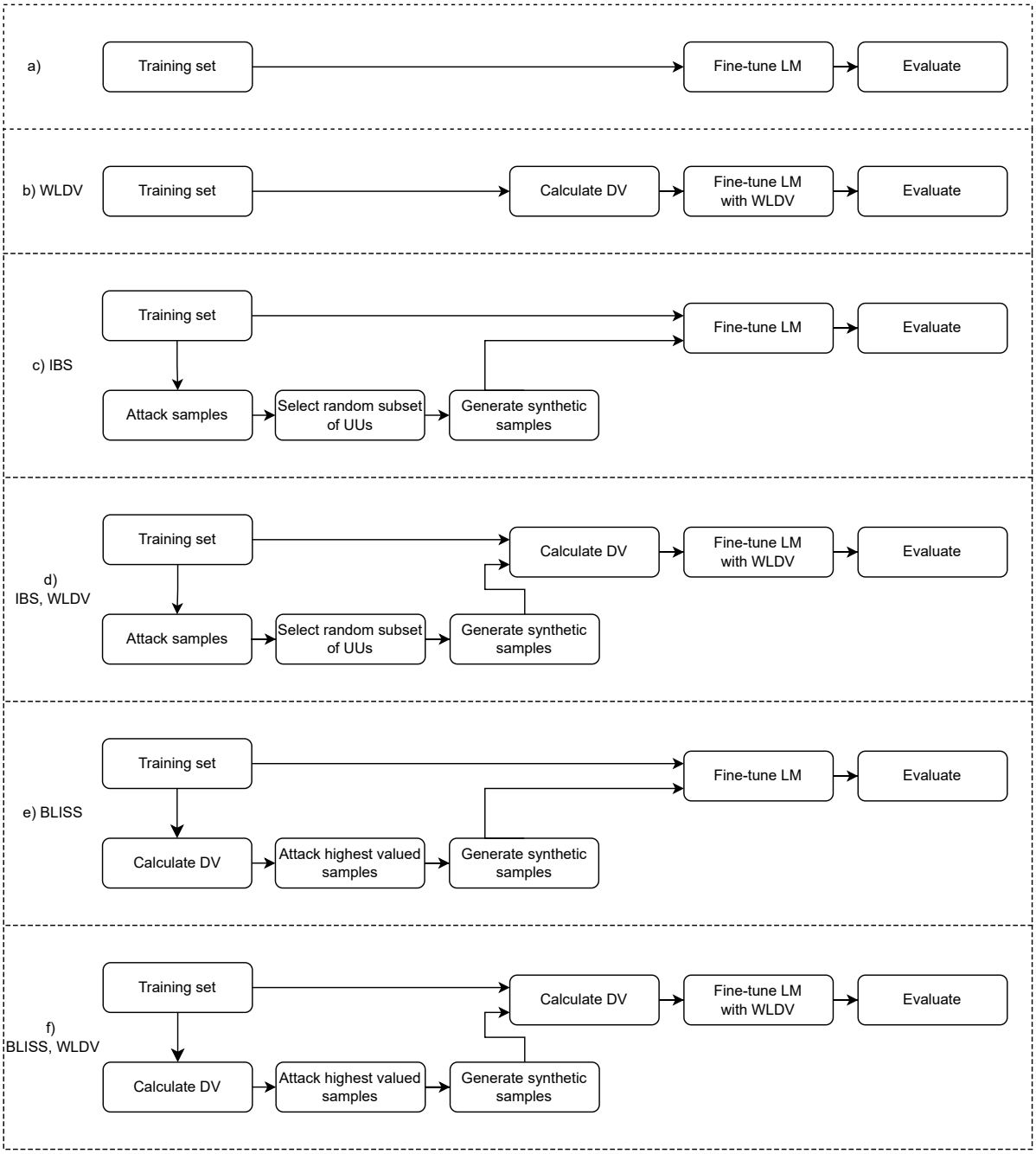


Figure 3.4: Four different combinations were compared to each other. a) Fine-tuning the LM using the original dataset, without additional synthetic samples. b) Fine-tune the LM within WLDV as weights, without additional synthetic samples. c) Extend the dataset by generating synthetic samples based on the highest-valued samples in the training set. d) The proposed method: Extend the dataset similarly to c, but calculate the DV again and use those DV to fine-tune the model using weighted loss.

Results

This chapter presents the results of method section 3. The goal was to investigate how synthetic data generation and data valuation could be used to mitigate blind spots in LM. This was investigates in two steps. First, using data valuation as a selection method for UUs to generate synthetic samples. Secondly, retraining the LM using weighted loss based on data values (WLDV). Following the approach of section 3.7, two selection methods have been compared, IBS and BLISS. When retraining the model, with a weighted loss based on data values (WLDV) and without the use of data values were compared. The chapter is divided into 6 sections, starting with the validation of data values in section 5.1. Section 5.2 compares the data values between the synthetic and original datasets. Section 5.3 investigates the effect on the reduction of UUs. Section 5.4 investigates the distribution of the confidences. At last, an investigation of reliability diagrams (section 5.5)

5.1. Validation of data values

Before the data values were used, we checked if the data values indeed represented the importance of the samples. To do this, a ”high/low” experiment was conducted. In the high/low experiment, a portion of the highest or lowest-valued samples were removed from the training set and the accuracy of the model was re-evaluated. It was anticipated that the model's accuracy would deteriorate when the highest-valued samples were excluded, as compared to utilizing the complete training set. This expectation was based on the assumption that the most valuable samples contribute significantly to the model's performance. On the contrary, when removing the lowest-valued samples, it was expected that the model's performance would either remain consistent or even improve. We conducted the experiment on synthetic samples generated using IBS and synthetic samples generated using BLISS. In Figures 5.1a-b, 5.1c-d and 5.1e-f, the result of the high-low experiment are shown for the IMDb, MRPC, and Email datasets respectively. As expected, in all three datasets the accuracy deteriorated when removing the highest-valued samples, and stayed equal when removing the lowest-valued samples. There is a difference between the accuracy drop when adding synthetic samples using IBS (left plots) and using BLISS (right plots). The MRPC and Email dataset resulted in an earlier drop in accuracy, which shifted from 15% to 9% in the Email dataset and 22% to 9% for the MRPC dataset. The drop in accuracy for the IMDb dataset stayed the same at both situations 28% removal.

The findings suggest that data valuation can effectively distinguish between samples of high and low importance across all three datasets, aligning with our expectations. However, adding synthetic samples using BLISS shows an earlier drop in accuracy in both the MRPC and Email datasets. The IMDb dataset does not show this behavior. This suggests that the model is more robust for removing samples using IBS. However, it is possible be that the samples of BLISS influenced the data values, such that more important samples had higher values and were removed earlier. Moreover, the aim was to mitigate blind spots, and a drop in accuracy does not necessarily indicate the measure of blind spot mitigation.

The Email dataset showed a short decrease in model accuracy after 5% sample removal and restored its accuracy after removing 15% of the samples. This is not an expected result, as we expected the accuracy not to deteriorate, or deteriorate more slowly. The exact reason why this drop occurred was not be found and is more elaborated upon in the discussion (chapter 6).

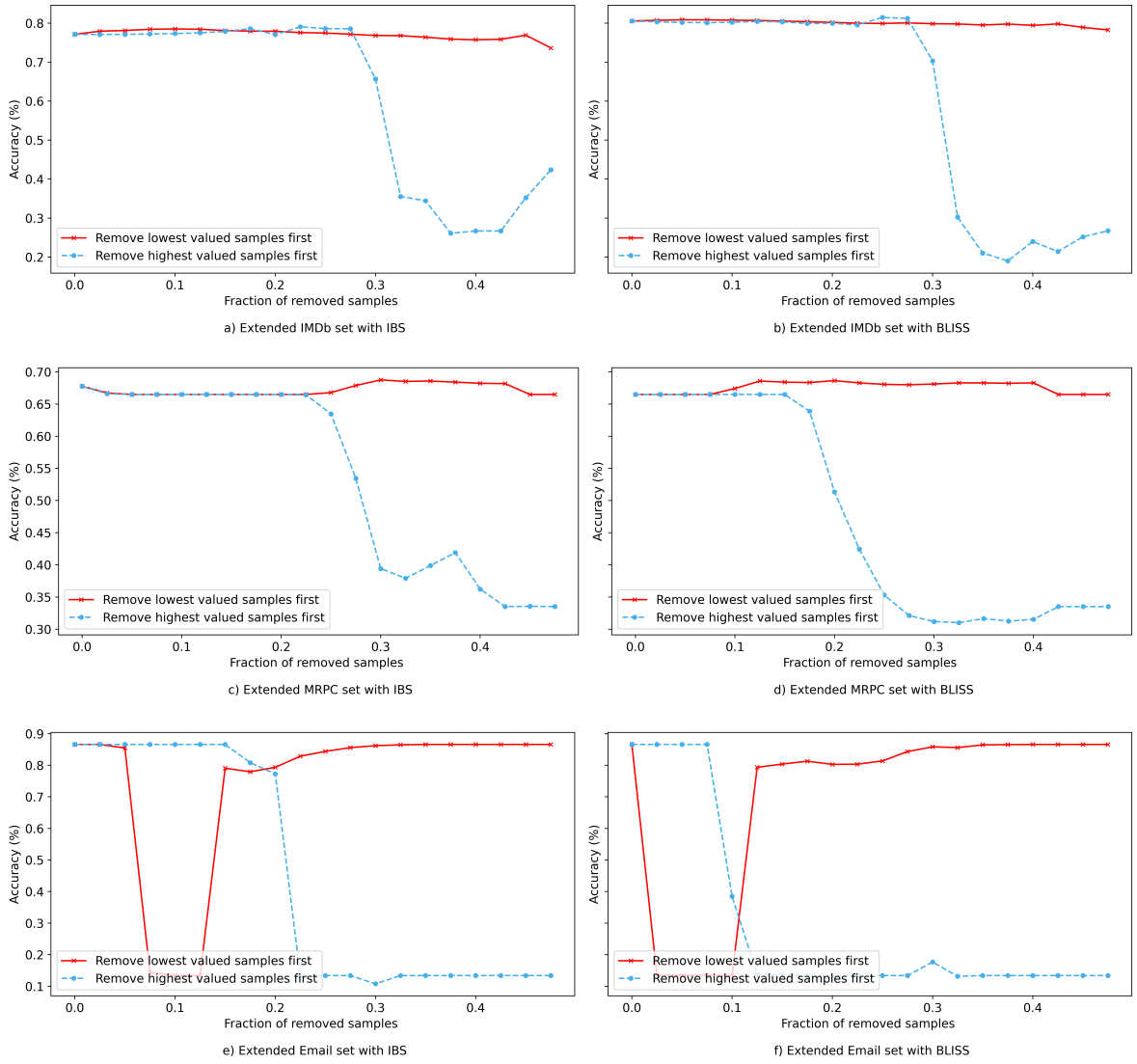


Figure 5.1: The high/low experiments were conducted on the IMDb dataset (a, b), on the MRPC dataset (c, d), and on the Email dataset (e, f). The left figures show the high/low experiment conducted on the dataset extended with synthetic samples generated using IBS. The right plots show the datasets extended using synthetic samples generated using BLISS

5.2. Synthetic data values

To evaluate the effect of adding synthetic samples generated using IBS or BLISS, we plotted the data values in a boxplot (Figure 5.2). In these figures, you can see the values given to the samples in both sets, compared to the original samples. Because the samples affect each other's valuation, we calculated the data values twice, once extending the training set with IBS and once with BLISS. The data values of the synthetic samples are shown on the left side, the right side contains the data values of the original samples.

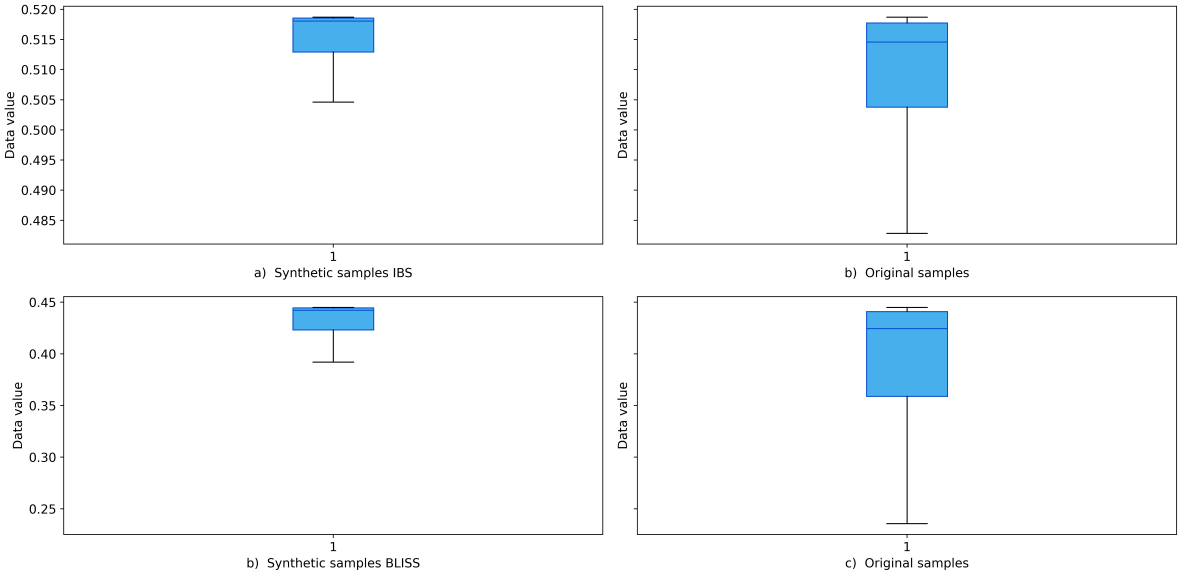
For the IMDb dataset, both the synthetic samples generated using IBS and BLISS contained higher values compared to the original samples. The maximum data values for IBS were the same with 0.51 for both the synthetic samples and original samples. For BLISS, the maximum values were almost similar, 0.45 and 0.44 for the synthetic and original samples respectively. However, the lowest-valued samples were different, with 0.513 and 0.504 for the synthetic samples generated using IBS and the original samples. And the minimum values of 0.42 and 0.36 for the synthetic samples generated using BLISS and the original samples. The distribution of data values using BLISS seem to be more skewed to the top, compared to IBS. The distribution of the original samples seems to be similar, although distributed in a different range. This may imply that the samples generated using BLISS are more valuable.

For the MRPC dataset, a clear difference is visible between the data values of IBS and BLISS. The samples generated using BLISS have, in proportion to the original samples, lower values compared to IBS. The samples of IBS follow the distribution of the original samples, but the medium value is with 0.02 slightly, higher valued. The lower-valued synthetic samples by BLISS are not necessarily a problem, as this could suggest that the originally underrepresented areas are now more common, and therefore less important for the model's accuracy. It could even indicate that a blind spot was mitigated.

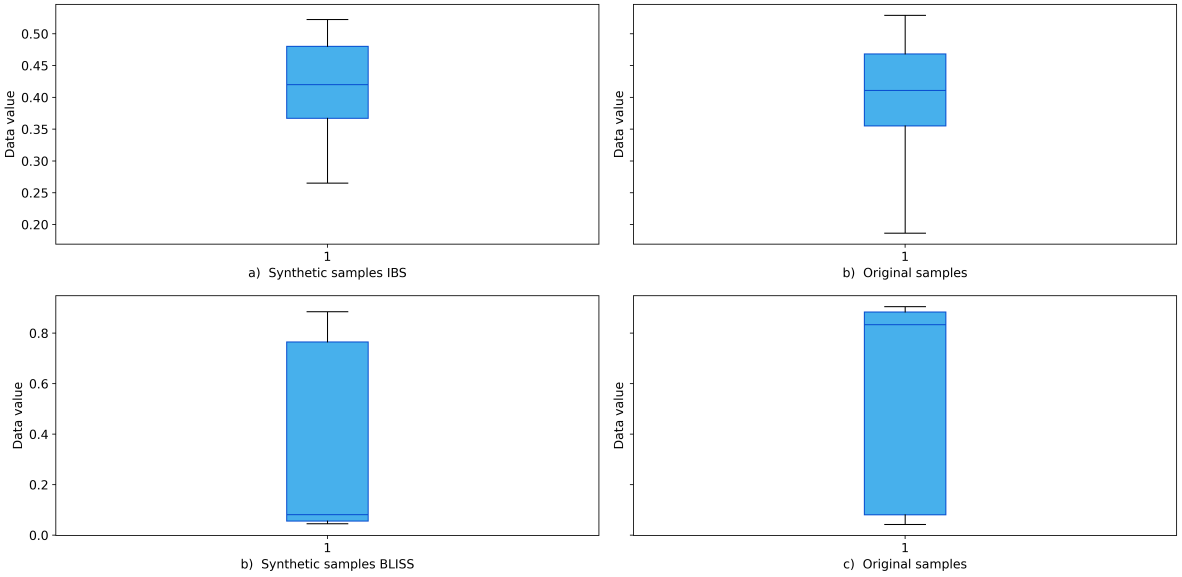
The distribution of the Email dataset shows a similar pattern as the MRPC dataset. The synthetic samples were only generated using BLISS because the previous study of IBS did not conduct this experiment. The samples are, similar to the MRPC dataset, lower compared to the original samples. This could indicate that the blind spot is now overrepresented in the training set, making them less important.

The difference between the IMDb, MRPC, and Email datasets could be caused by the difference in the validation set sizes between IMDb, MRPC, and Email. The validation set sizes are 500, 408, and 317 samples for the IMDb, MRPC, and Email datasets respectively. This is, in ratio to the training set, the smallest validation set. To investigate the assumptions further, the next section takes a deeper look at the effect of the number of UUs.

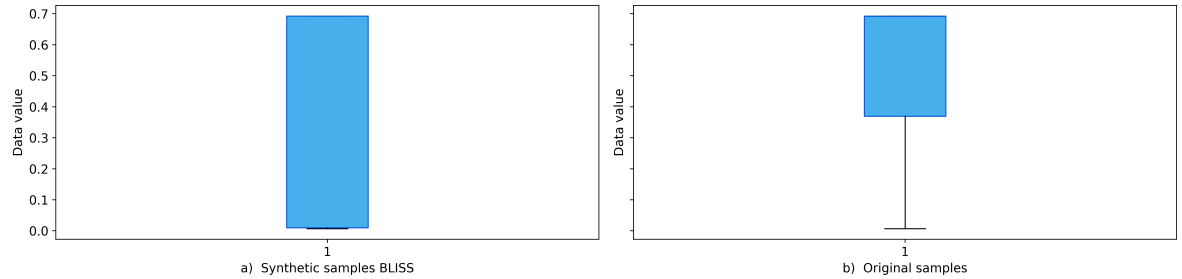
An important observation is that the magnitude of the data values between IBS and BLISS on the same set can differ. E.g, for the IMDb dataset, the range of data values is between $[0.52, 0.55]$ for the dataset extended with samples of IBS, whereas the IMDb dataset extended with samples of BLISS is in the range of $[0.45, 0.25]$. This behavior has not been investigated in related studies, therefore, we can't immediately know which expectations we should have. However, previous work does indicate that data values should behave according to certain rules. These desiderata do indicate the effect when a subset of data has changed. The difference is discussed further in the discussion section 6.



(a) IMDb



(b) MRPC



(c) Email

Figure 5.2: Comparing the data values between the synthetic samples generated by BLISS and the original samples of the dataset. Figures (a,b) are the IMDb dataset. Figures (c,d) are the MRPC dataset. Figures (e,f) are the Email dataset. *Please note that the scales are not similar between the boxplots. Only on the horizontal axis.*

5.3. Mitigation of Blind spots

As described in the method section 3.7, we evaluate our method using 6 combinations on three datasets. Three datasets were IMDb, MRPC, and Email. The QNLI dataset was excluded from the analysis due to the absence of appropriate data values. A detailed discussion regarding this limitation can be found in Section 6. The Email dataset did not include the synthetic samples generated using IBS, as Lippmann et al. did not conduct that dataset. To recap, the combinations were:

1. As a base case, the LM was only retrained on the original dataset, without any synthetic samples or using data values.
2. Retraining the LM on the extended set with synthetic samples generated using BLISS.
3. Retraining the LM on the extended set with synthetic samples generated using IBS.
4. Retraining the LM using WLDV.
5. Retraining the LM using IBS, and including WLDV
6. Retraining the LM using BLISS, and including WLDV

A more detailed explanation of each combination can be found in section 3.7. The following sections discuss the results for each dataset.

5.3.1. IMDb

Following the BLISS method, described in section 3.5, 500 synthetic samples were generated for the IMDb dataset. In table 5.1 the results for the IMDb dataset are shown. The accuracy for each combination was about the same ($\pm$ 0.2%). Looking at the number of UUs after perturbation, the combination IBS, WLDV had the least number of UUs, even though the accuracy after perturbation was not the lowest. Since the accuracy was lower, this meant that more samples were classified.

The results in table 5.1 indicated that the accuracy before the adversarial attack was relatively consistent across all combinations, ranging from 92.1% to 92.3%. However, after the adversarial attack, the accuracy varied more significantly. The IBS combination achieved the highest accuracy after the attack at 24.9%, while the ”BLISS, WLDV” combination had the lowest accuracy at 19.4%. In terms of UUs after perturbation, the IBS, WLDVcombination produced the fewest UUs with 9,851, while BLISS generated the most with 10,404.

Incorporating samples generated by BLISS negatively affected the model’s accuracy and raised the number of UUs. This could potentially be due to the samples focusing only on a particular blind spot. The combination of BLISS and WLDV resulted in a smaller number of UUs compared to only using WLDV. The combination showed a complementary effect, where the synthetically generated samples using BLISS addressed certain blind spots, while the sample weights (WLDV) ensured that the model did not overemphasize these blind spots at the expense of other important patterns or information. The combination of IBS and WLDV reduced the most UUs most likely because the samples of IBS were based on a random selection of UUs. This might create a more diverse set of samples. Only using IBS appeared to be more resilient in maintaining accuracy after an attack.

Some generated outputs contained two or more samples, while only 1 sample was required. If this happened, the last sample was used and the others were disregarded. This happened twice times for the IMDb dataset.

5.3.2. MRPC

The MRPC dataset, with 3,668 samples, was considerably smaller than the IMDb dataset, which had 24,500 training samples. The test and validation set contained 1,725 and 408 samples respectively and 63 additional synthetic samples were generated. In table 5.2 the results on the MRPC dataset for each combination are shown.

Table 5.1: The results after retraining the LM for each combination for the IMDb dataset. It shows the accuracy before the adversarial attack, accuracy after the adversarial attack, and the number of UUs after perturbation using DeepWordBug.

Combination	Accuracy (%)	Confidence (mean, std)	Accuracy after perturbation (%)	UUs (#)
only	92.2%	77.8% $\pm$ 5.9%	24.1%	10,024
WLDV	92.1%	75.3% $\pm$ 6.6%	24.0%	9,902
IBS	92.3%	78.2% $\pm$ 5.9%	**24.9%**	9,978
IBS, WLDV	92.3%	75.8% $\pm$ 6.2%	21.5%	**9,851**
BLISS	92.2%	75.8% $\pm$ 6.1%	19.6%	10,404
BLISS, WLDV	92.2%	74.7% $\pm$ 6.5%	19.4%	9,906

Comparing the results to those obtained from the IMDb dataset, it was noticeable that the accuracy before the adversarial attack varied more significantly across the combinations on the MRPC dataset, from 69.9% to 81.7%. This seemed mainly to be caused in two cases, after retraining using WLDV and "BLISS, WLDV". This could indicate that the data values played a role in this accuracy drop. In both cases, the number of UUs after perturbation was the lowest, with 337 and 313 UUs. To investigate the accuracy drop, we added the average confidence. Those were for both cases lower (0.73% and 0.75%) compared to the other combinations (0.83% to 0.87%). The drop in the number of UUs was therefore caused by the model having less overall confidence, instead of mitigating blind spots. This indicated that only using the number of UUs was not a perfect metric to indicate blind spots, but should always be seen together with the average confidence.

In a similar manner to the IMDb dataset, incorporating synthetic samples generated by BLISS raised the number of UUs (421). When combining BLISS with WLDV, decreased the number of UUs (313), which was also similar to the IMDb dataset. However, we were hesitant to conclude that this was a consistent pattern, as the drop in average confidence was more likely the cause behind this reduction in UUs.

Another observation is the increase in the number of UUs after adding WLDV to IBS, even though the accuracy before the attack, accuracy after the attack and the average confidence were similar. This was in contrast to what we have seen in the IMDb dataset. To better understand the reasons behind this difference, further analysis and experimentation was needed, focusing on the interaction between the datasets, the methods, and the LM.

Table 5.2: The results after fine-tuning the LM for each combination for the MRPC dataset. It shows the loss, accuracy, and the number of UUs before an adversarial attack.

Combination	Accuracy (%)	Confidence (mean, std)	Accuracy after attack (%)	UUs (#)
only	81.0%	83.3% $\pm$ 15.2%	15.9%	387
WLDV	69.9%	75.4% $\pm$ 12.1%	18.9%	337
IBS	81.7%	84.0% $\pm$ 14.7%	15.6%	371
IBS, WLDV	81.5%	83.6% $\pm$ 14.7%	15.7%	387
BLISS	81.5%	82.9% $\pm$ 15.4%	16.3%	421
BLISS, WLDV	67.4%	73.3% $\pm$ 11.3%	15.5%	**313**

5.3.3. Email

The Email dataset was the largest, containing 31,716 training samples, 5,574 test samples, and 317 validation samples. The Email dataset was meant to investigate the effect of OOD data. The training set consisted of email samples, but the test consisted of SMS spam samples. There were 630 synthetic samples generated, from which 5 samples were discarded because they did not contain a reason and

Table 5.3: The results after fine-tuning the LM for each combination for the Email dataset. It shows the loss, accuracy, and the number of UUs before an adversarial attack.

Dataset	Accuracy (%)	Confidence (mean, std)	Accuracy after attack (%)	UUs (#)
only	37.1%	95.5% ± 9.8%	5.3%	1,684
WLDV	34.3%	95.8% ± 9.7%	**6.3%**	**1,425**
BLISS	34.2%	96.3% ± 9.1%	4.9%	1,509
BLISS, WLDV	32.2%	95.6% ± 9.8%	3.4%	1,446

label. In table 5.3 the results on the Email dataset are shown.

For the Email dataset, the accuracy before the adversarial attack was relatively low across all combinations, ranging between 32.2% to 37.1%. This was expected because the LM was trained on email spam classification, but evaluated on SMS spam classification. After the attack, the accuracy dropped significantly for all combinations, with WLDV showing the highest accuracy at 6.3%. In terms of UUs before perturbation, retraining using WLDV produced the fewest UUs with 1,425, while only using the Email dataset generated the most with 1,684 UUs. The mean confidence values were relatively high and consistent across all combinations, ranging from 95.5% to 96.3%. These findings were expected and suggested that the performance of the LM on the Email dataset was positively affected by adding synthetic samples using BLISS. It shows that BLISS could target samples that looked like SMS samples. Consistent over all datasets, combining BLISS and WLDV had fewer UUs compared to only using BLISS, and only retraining using WLDV resulted in fewer UUs compared to BLISS and WLDV.

5.4. Confidences

In the previous section, the number of UUs was based on a confidence threshold of $\tau \geq 0.65$. This section delves deeper into the distribution of confidences by plotting the distributions on histograms. This can give more detailed information. The subsequent paragraphs discuss these distributions for each dataset. The figure shows the confidence distribution of successful perturbed samples. I.e., samples that were perturbed, and the model predicted an incorrect label. The confidence was always $c \geq 0.5$, because it was a binary classification problem. The red line indicated the UU threshold $\tau \geq 0.65$. The goal was to lower the overall number of samples in the diagram since that would indicate better accuracy and to push samples from high confident predictions to lower confident predictions. The figures show the confidence distribution of successful perturbed samples. I.e., samples that were perturbed, and the model predicted a different (incorrect) label.

5.4.1. IMDb

Upon retraining solely on the IMDb dataset, there were 17,385 incorrectly classified samples, with 10,024 samples classified as UUs and 7,361 as known unknowns. The number of samples generally decreased as confidence increased, but there was a rise in samples with confidence $c \geq 0.9$, totaling 3,503 samples. A slight reduction in the overall number of incorrectly classified samples (17,204) was observed when extending the training set with synthetic samples generated using IBS, and also the number of UUs dropped from 10,024 to 9,978. Extending the dataset using synthetic samples generated by BLISS resulted in the highest number of incorrectly classified samples (18,529), from which 10,404 UUs. Adding WLDV decreased the number of UUs in all three settings, where the lowest number of UUs was achieved by combining IBS and WLDV, having 9,851 UU, which was a drop of 1,7% compared to only using the IMDb dataset. However, it seemed that the synthetic samples of BLISS mainly reduced the highest confident samples. Increasing the threshold to $\tau = 0.9$, we call these $\text{UU}_{0.9}$, the combination of BLISS and WLDV performed the best. reducing the number of $\text{UUs}_{0.9}$ to 2792. Compared to just

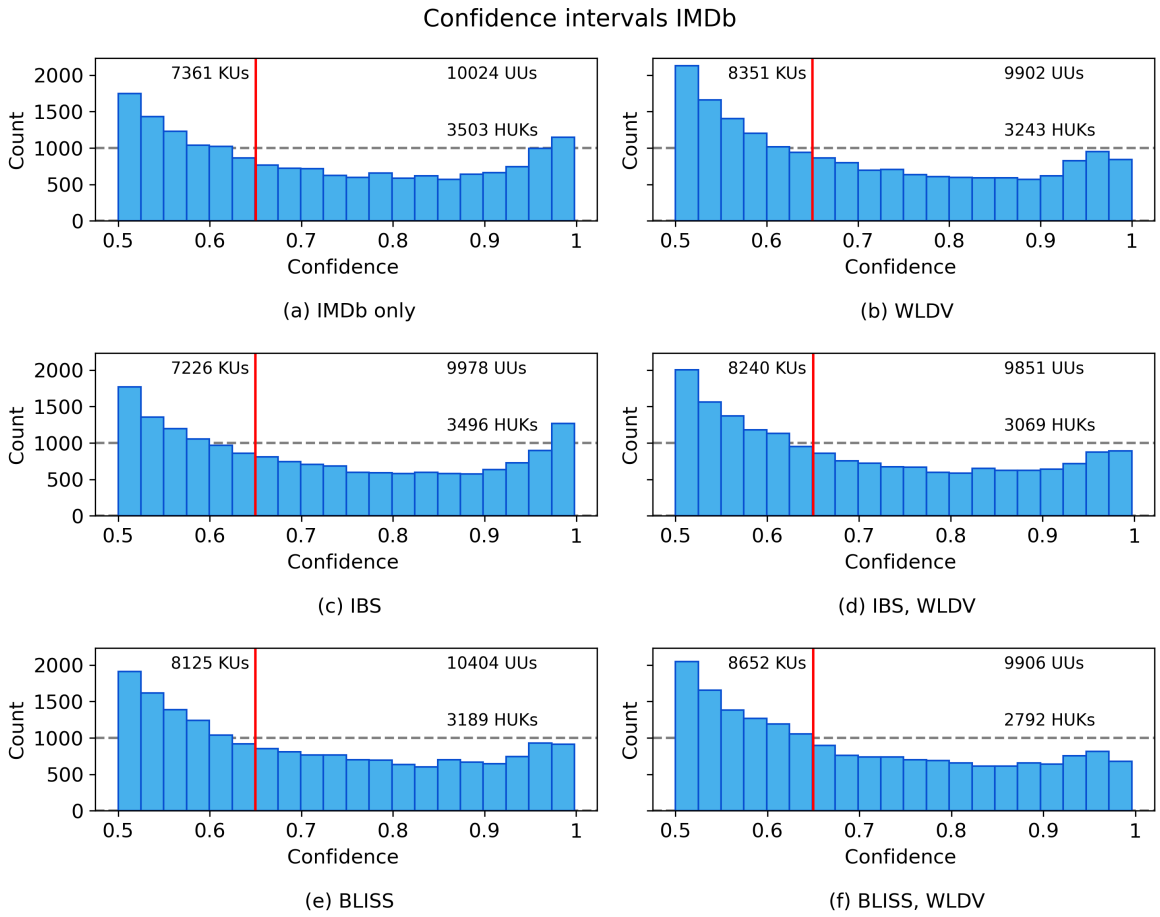


Figure 5.3: Confidence intervals when looking at incorrectly classified samples after perturbation on the IMDb dataset. The red line indicates confidence $\tau = 0.65$. For the HUU, the threshold $\tau = 0.8$ was selected.

using the IMDb dataset (3503 $UUs_{0.9}$), this is a reduction of 20.3%.

The results suggested that WLDV was effective in reducing the number of UUs in a language model, decreasing both the overall number of UUs. Although adding synthetic samples using BLISS did not reduce the total number of UUs at $\tau = 0.65$, it resulted the least number of UUs when the threshold was increased to $\tau = 0.9$. This could indicate that BLISS's samples were effective in targeting some blind spot samples that previously caused very high confidence predictions ($c \geq 0.9$). In summary, the combination of IBS and WLDV had the lowest number of UUs, but looking at confidences higher than 0.9, the combination BLISS and WLDV has the lowest number of samples. This highlights the importance of evaluating the impact of synthetic samples on different confidence levels. The threshold of a UU $\tau \geq 0.65$ was chosen because originally Lakkaraju et al. chose this threshold (Lakkaraju et al. 2017). However, questions may be asked about the specific value, as they did not argue why it was 0.65. Attenberg et al. did not define a threshold, but kept it more broadly and defined it as "a high difference between the expected misclassification cost and actual misclassification cost" (Attenberg, Ipeirotis, and Provost 2015).

A silent detail is that extending the training set with IBS had the highest accuracy, having 17,204 incorrect classified samples. However, looking at the number of UUs, adding WLDV to IBS, has the lowest number of UUs, 9,851 compared to 9,978. This indicates that even though the accuracy was lower, more samples were classified as known-unknowns. This is good, as this indicates that more UUs moved to the known unknown space.

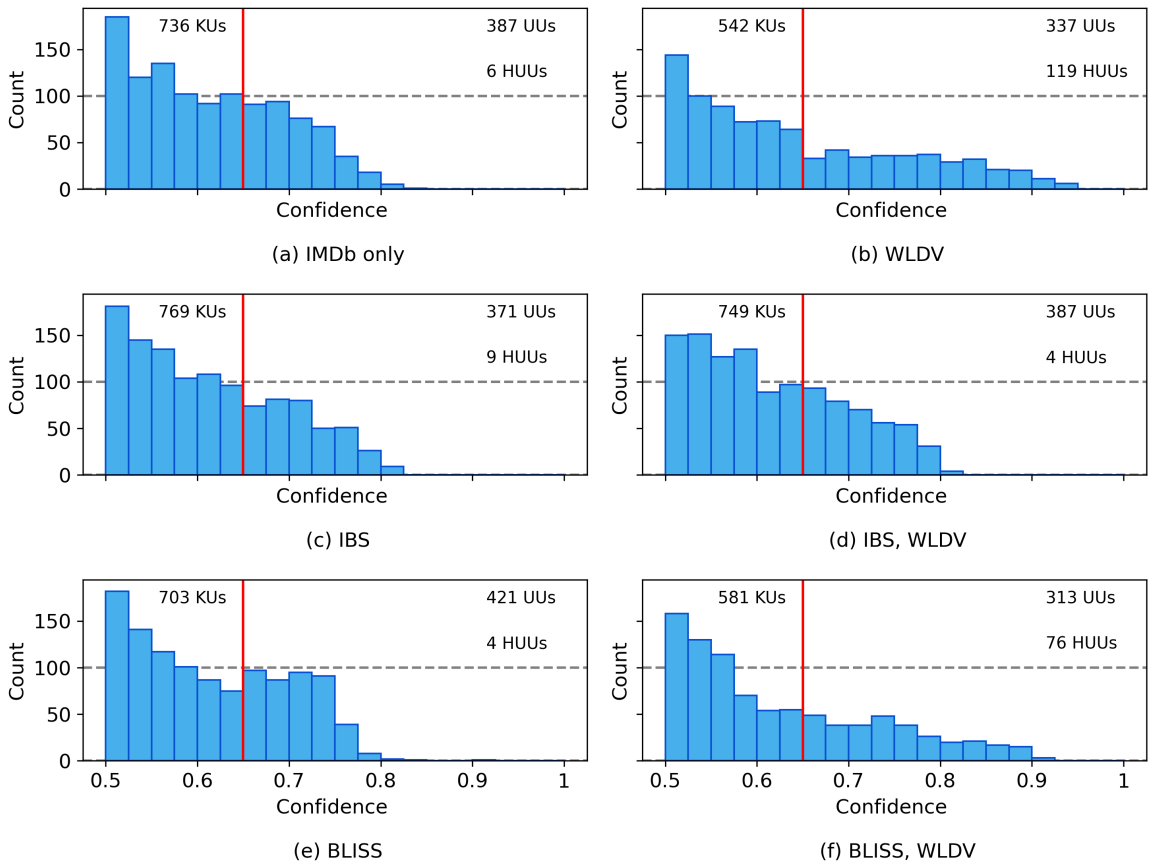


Figure 5.4: Confidence of incorrectly predicted samples for the MRPC dataset. The red line indicates confidence $\tau = 0.65$.

5.4.2. MRPC

For the MRPC dataset, a more fluctuating trend is observed compared to the IMDb dataset. Retraining solely on the MRPC dataset resulted in 1,123 incorrectly predicted samples, with 387 classified as UUs. Adding synthetic samples generated using IBS led to a minor decrease in UUs (from 387 to 371), but the total number of incorrectly classified samples increased (from 1,123 to 1,140), with no significant change in the distribution. Incorporating synthetic samples using BLISS appeared to be disadvantageous for the number of UUs, increasing it to 421, while the total number of incorrectly labeled samples was 1,124. The largest reduction in UUs, with $\tau = 0.65$, was observed by combining BLISS and WLDV, with 313 samples. This was a reduction of 19% compared to just using the IMDb dataset. However, we saw a more spread-out distribution of samples when adding WLDVat a confidence $c \geq 0.8$. When increasing the threshold to $\tau = 0.8$, we call these $\text{UU}_{0.8}$, the best performance was gained by combining IBS and WLDVor BLISS, with both having 4 $\text{UUs}_{0.8}$. However, we must

Applying WLDV had a noticeable impact on the distribution of confidence levels with a confidence $c \geq 0.8$. The addition of WLDV resulted in a more spread-out distribution of confidence levels when just adding WLDV to the IMDb dataset (199 $\text{UUs}_{0.8}$), and combining BLISS and WLDV(76 $\text{UUs}_{0.8}$). Interestingly, this spread was not observed when incorporating synthetic samples generated using IBS.

The difference in the impact of IBS and BLISS could be related to the range of data values in each set. As shown in Figure 5.2, the range of data values for IBS is between $[0.23, 0.52]$, while the range for BLISS is between $[0.01, 0.9]$. This wider range in BLISS might account for the stronger influence on the confidence distribution.

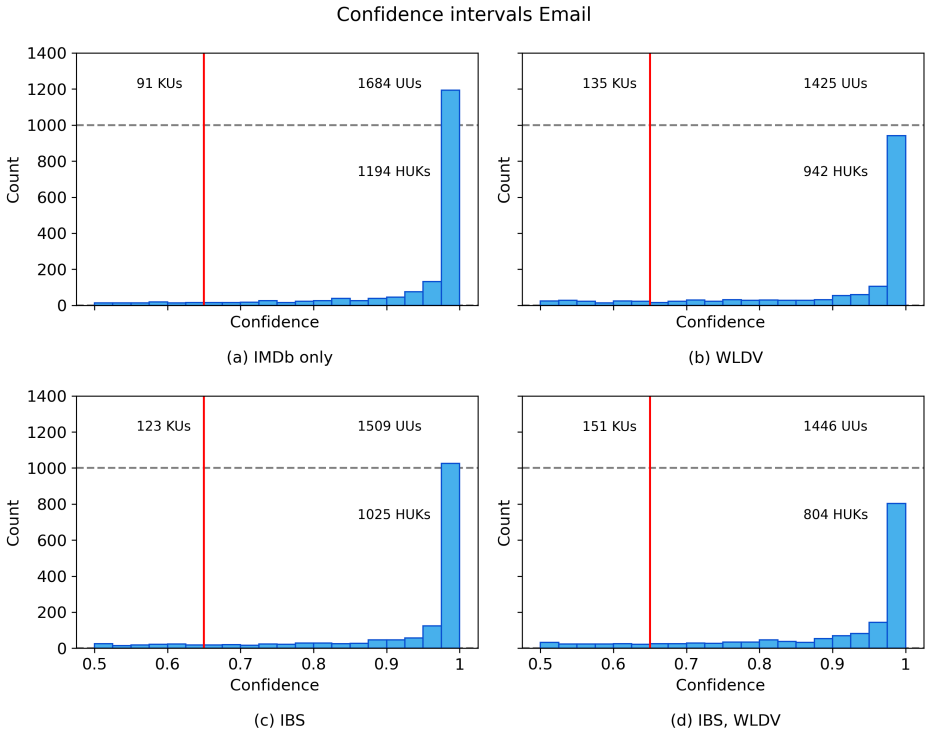


Figure 5.5: Confidence of incorrectly predicted samples for the Email dataset, the red line indicates $\tau = 0.65$. It shows a decreasing peak of incorrect classified samples with more than 100% confidence. (a) Represents only the Email dataset, without synthetic samples. (b) Represents retraining using WLDV. (c) Represents adding synthetic samples using BLISS(d) Represents synthetic samples generated using BLISS and retraining using WLDV.

5.4.3. Email

Examining the confidence levels in the Email dataset, as depicted in Figure 5.5, showed a peak in 100% confident predictions. This peak may be attributed to the class imbalance of the SMS dataset, as the Email dataset was equally split into spam and not spam, whereas the SMS dataset was primarily composed of spam samples.

The highest number of UUs was observed after retraining the LM solely on the Email dataset (1,684). Extending the training set with synthetic samples using BLISS led to a decrease in UUs to 1,509. In both cases, the addition of WLDV resulted in a reduction in the number of UUs, from 1,684 to 1,425 (15% reduction) and from 1,509 to 1,446 (4,2% reduction), respectively. The decrease in the number of UUs indicated that the methodology was effective. Similarly to the IMDb and Email dataset, increasing the threshold to $\tau = 0.9$ changed the number of UUs. Now, the combination of BLISS and WLDV had the least number of $\text{UUs}_{0.9}$ (804) compared to using only WLDV (942 $\text{UUs}_{0.9}$). This was a decrease of 32% compared to the number of $\text{UUs}_{0.9}$ when using just the IMDb dataset (1194).

5.5. Reliability diagrams

With the lack of a common metric to measure blind spot mitigation, we tried to investigate if the reliability curves could be a useful measurement. The reliability curves have been employed in related concepts, such as confidence calibration but not yet on blind spot mitigation.

A reliability curve, also known as a reliability diagram, is a visual representation of the relationship between the predicted confidence of a model and its actual accuracy. The curve plots the empirical probability (y-axis) against the predicted probability (x-axis). In an ideal scenario, a perfect model would

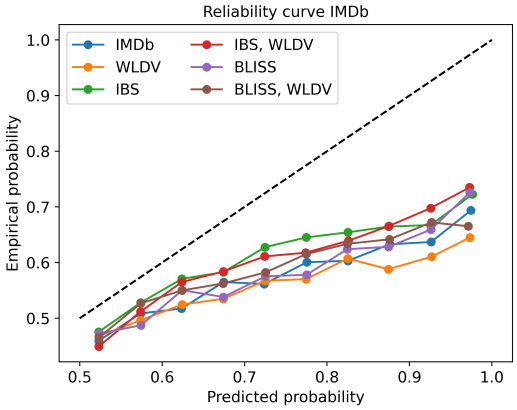
have a reliability curve that aligns with the diagonal line, indicating that its predicted confidence matches its actual accuracy. I.e., if the confidence of a prediction was 80%, this would imply that 80% of those predictions were accurate. By comparing the reliability curves between the different combinations, we could assess which effect the synthetic samples and WLDV had on the reliability of the confidence. More details about the reliability curve were discussed in the related work section 2.1.4.

Figure 5.6a shows the reliability curve for the IMDb dataset. The curves appeared quite similar to one another. All confidence curves were below the optimal reliability curve (black dashed line). This discrepancy suggested that the confidence was not well calibrated. The combination of IBS, WLDVand only adding synthetic samples using IBS seemed to be closest to the optimal reliability curve. Yet it was difficult to determine whether this proximity represented a meaningful difference compared to the other combinations since all combinations were about the same distance from the optimal reliability curve.

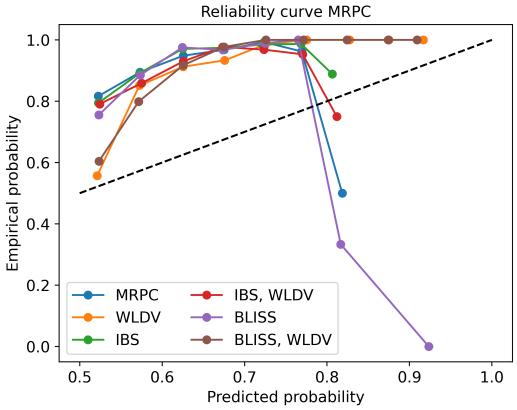
The reliability curve for the MRPC dataset (Figure 5.6b) showed a big drop for the IBS and ”IBS, WLDV” combinations. The drop was caused by the limited number of samples that occurred with these high probabilities. Therefore, the calibration was influenced by just a few samples. No conclusions should be made based on the drop.

The reliability curve of the Email dataset (Figure 5.6c) showed a downward trend, indicating first an underconfident model, moving to overconfidence. This could be caused by the class imbalance, which was a similar reason for the high peak in 100% confident predictions discussed in the previous section.

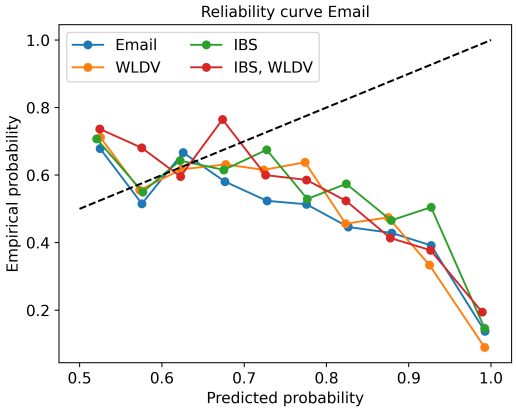
We can conclude that the reliability curve was not an effective measure for blind spot mitigation. It does not show any improvement between combinations and, because the MRPC dataset has few samples, showed a drop.



(a) IMDb



(b) MRPC



(c) Email

Figure 5.6: The reliability curves for the IMDb, MRPC, and Email datasets. The curves in the IMDb and Email datasets looked like each other. The curves in the MRPC dataset showed a drop for the IBS and IBS, WLDV combinations. However, these were based on a limited number of samples (4 and 8). Since the curves looked like each other, the reliability curves are not a good metric to measure blind spot mitigation.

Discussion, Limitations, and Recommendations

In reviewing the literature, there was a need for a scalable solution to address blind spots in large language models (LMs). The recent work by Lippmann et al. introduced a cost-effective approach to mitigating blind spots using synthetic sample generation (Lippmann, Spaan, and Yang 2024). They generated synthetic samples by selecting a random subset of unknown unknown (UU) predictions. This raised questions about the effectiveness of the random selection method. In this study, we proposed a selection method based on the causes of blind spots, such causes are out-of-distribution (OOD) data, bias, covariate shift, or unseen data (Attenberg, Ipeirotis, and Provost 2015; Chung et al. 2019). We used data valuation, a research field focused on valuing samples, which can assign higher values to OOD or biased data and detect noise (Sim, Xu, and Low 2022; Yoon, Arik, and Pfister 2020). In this study, we tried to answer the research question, "How can data generation and data valuation be used to mitigate blind spots in large language models?"

We integrated data valuation and data generation in two steps. First, we used data valuation as a selection method for UUs. Using high-valued samples, we generated synthetic samples. This was called BLISS. We compared BLISS to Lippmann et al.'s random selection method, abbreviated as IBS. In the second step, we retrained the LM using Weighted Loss based on Data Values (WLDV) and evaluated both steps independently to determine their contributions.

Our findings indicated that adding WLDV always improved the reduction of UUs, regardless of whether the dataset was extended with synthetic samples or not. It showed the biggest reduction in the Email dataset with 15% while maintaining the same level of accuracy. Regarding synthetic sample generation, on the IMDb dataset combining synthetic samples generated using IBS with WLDV outperformed combining BLISS with WLDV in reducing the number of UUs when considering the confidence threshold $\tau = 0.65$ with a reduction of 1.7% compared to only using the IMDb dataset. On the MRPC dataset, the combination of BLISS and WLDV performed best, reducing the number of UUs by 20%.

The synthetic samples generated using BLISS seemed to affect very high confidence predictions ($c \geq 0.9$) more compared to the samples of IBS. Increasing the threshold for the IMDb and Email datasets to $\tau = 0.9$ resulted in a more significant drop. Here, the combination between synthetic samples generated using BLISS in combination with WLDV, showed a drop of 20.3% on the IMDb and 32.7% compared to just using the IMDb and Email respectively. This drop was significantly higher compared to using the sampled generated using IBS, which achieved a 12.4% drop on the IMDb dataset and a 21.1% drop on the Email dataset. Although the number of UUs reduced when the confidence $c \geq 0.9$ on the IMDb and Email dataset, the MRPC dataset did not show a similar pattern. However, in the MRPC dataset, most predictions had a confidence lower than 0.8. The results showed that data valuation can be used to systematically reduce the number of UUs. The results emphasized that it was important to carefully determine the UU threshold τ. The threshold of $\tau = 0.65$ followed the definition of Lakkaraju et al., however, others such as Attenberg et al. (Attenberg, Ipeirotis, and Provost 2015) did not have a fixed threshold and said that a prediction is UU when the estimated misclassification cost was low, but the true classification cost was high.

What was surprising was that only adding synthetic samples generated using BLISS increased the number of UUs when using the threshold $\tau = 0.65$, but did reduce the number of samples when the threshold was set to $\tau = 0.9$. This study confirms that extending the training set using synthetic samples

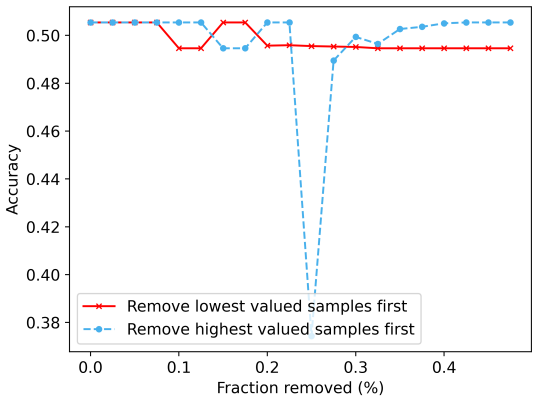


Figure 6.1: The QNLI dataset showed no difference when removing high or low data values. This implied that the data valuation method was not working, since removing high or low-valued samples did not impact the accuracy of the model.

generated based on UUs, reduced blind spots in LMs.

6.1. Limitations

Data valuation was employed across various datasets. While this approach resulted in expected outcomes on the high/low experiments on the IMDb, MRPC, and Email datasets, this did not yield the expected outcome on the QNLI dataset. One potential explanation could be attributed to the more advanced task of the QNLI dataset, determining whether a sentence contains the answer to a question (Wang et al. 2018). It could also be caused by the large size of the dataset. Since the high/low experiment showed no correlation between data values and accuracy, see Figure 6.1, the QNLI dataset was excluded from further analysis.

During the generation of data for the Email dataset, the quality of the generated samples was somewhat noisier compared to the other datasets. OpenAI's *text-davinci-003* model produced some instances, 10 in total, containing two or more samples, labels, and/or reasonings without being explicitly prompted. Additionally, the model occasionally trimmed samples by only mentioning the perturbation it would apply. In cases where multiple samples were generated, only the last sample was considered. If the model only proposed a perturbation, the sample was left unchanged. Although a small fraction of the samples were generated incorrectly, this issue affected only a minority of the total samples.

Another limitation was the maximum input length of 1024 tokens imposed by the model. As a result, some input sentences in the Email dataset were truncated, affecting both the original and perturbed sentences. Consequently, it is possible that some perturbed tokens were removed during trimming. However, this limitation only affected 5 samples.

During the calculation of data values, we observed variations between runs with identical settings. These variations in data values could potentially influence the outcomes. To ensure reproducible results despite the stochastic behavior, we opted to use a fixed seed. An empirical study revealed that while the magnitude of data values could change, the overall order of the samples remained relatively consistent. This limitation had minimal consequences, as we selected the highest-valued samples for our analysis. It could have had an impact when retraining the model, as the weights may differ. To assess the effect of these variations, each experiment could be conducted multiple times. We leave this investigation for future research, as the primary focus of this study was to explore the relationship between data values and blind spot mitigation, which can be demonstrated even under small variations.

The research design was limited to classification tasks within the text-based domain. It did not expand to different tasks, such as text generation, or domains such as image, speech, or video. Besides this, we

only used the adversarial attack method DeepWordBug (Gao et al. 2018). This led to a limited diversity in the generated synthetic samples, with a focus on character-level perturbations. Notwithstanding this limitation, DeepWordBug is a state-of-the-art perturbation method and was also used in other studies (Lippmann, Spaan, and Yang 2024). It is furthermore relatively easy to include other perturbation methods.

6.2. Recommendations

For future work, it is recommended to incorporate different tasks and domains. The calculation of data values can be easily adapted by replacing the predictor model with a task or domain-specific model. However, synthetic sample generation requires careful consideration, as the adversarial attack method should be tailored to the specific task and domain, and the generation process should produce different outputs.

Another potential improvement is sorting the samples before retraining the model. Since the LM was retrained in batches of 8 with an average loss, there is a risk of averaging out low and high data values. Sorting samples from high to low (or vice versa) before fine-tuning could potentially enhance the model's performance.

Future studies could explore different data valuation methods to investigate their impact on the results. As the data valuation method influences the outcome, comparing methods like the Distributional Shapley value (Ghorbani, Kim, and Zou 2020) or the Data Shapley value (Ghorbani and Zou 2019) could provide valuable insights. The current implementation in this study can be easily modified to replace the data valuation method with one of the supported methods in the Opendataval package (Jiang et al. 2023).

One promising direction for future research is the development of a standardized method to assess blind spot coverage in language models. This could entail creating a benchmark dataset that encompasses a wide range of blind spots, along with a consistent set of evaluation metrics. These metrics should take into account both confidence and accuracy as essential components.

Lastly, a natural progression of this research would be to establish a stronger connection between the calculation of data values and blind spot mitigation. A potential research question in this area could be: "How can data values be calculated specifically for blind spot mitigation?" This investigation could lead to more targeted and effective strategies for addressing blind spots in language models, however, a good benchmark or metric is likely required to accomplish this.

Conclusion

This research aimed to investigate a scalable approach for mitigating blind spots in large language models (LMs) by leveraging data valuation (DV) and synthetic data generation. Previous studies relied on crowd workers or oracles to address blind spots, which are not scalable solutions. However, the recent work by Lippmann et al. introduced a scalable system that generated synthetic samples based on unknown unknowns (UUs) (Lippmann, Spaan, and Yang 2024). We extended their approach, by looking at the underlying causes of blind spots, such as out-of-distribution (OOD), bias, or covariate shift. Our method utilized data valuation, a research field that values the importance of samples, enabling the detection of low-quality data, noise, or OOD (Sim, Xu, and Low 2022; Yoon, Arik, and Pfister 2020). We investigated how data valuation and data generated could be combined, to mitigate blind spots.

We introduced a two-step method to mitigate blind spots. First, we valued each sample using data valuation and selected the highest-valued samples. Using those samples, we generated synthetic samples. This is referred to as BLISS. We compared these to synthetic samples generated based on randomly selected samples, referred to as IBS. The synthetic samples were used to extend the training set. In the second step, we used data values to retrain an LM using Weighted Loss based on Data Values (WLDV).

We evaluated our proposed approach on four distinct text classification tasks. For each task, we examined the effectiveness of extending the training set using synthetic samples generated by our proposed method, BLISS, as well as the method proposed by Lippmann et al., IBS. Additionally, we evaluated the performance of the LM when retrained using our WLDV technique, as compared to retraining without WLDV.

Regardless if the training set was extending using BLISS or IBS, retraining the LM using WLDV showed a consistent reduction in the number of UUs, having the biggest reduction on the Email dataset with 15%, while maintaining the same level of accuracy. Comparing the synthetic sample generation based on BLISS or IBS, it depended on the threshold to determine if a prediction was high confident. Using a high confident threshold $\tau = 0.9$, the combination of BLISS and WLDV outperformed the combination of IBS and WLDV, reaching a reduction of UUs with 20.3% on the IMDb and 32.7% on the Email dataset, compared to 12.4% and 21.1% for the combination of IBS and WLDV. In the MRPC dataset, the combination BLISS and WLDV outperformed the combination IBS and WLDV when the threshold was set at $\tau = 0.65$, reducing the number of UUs to 19.1%, compared to 13.9%.

The findings suggested that data valuation can be used to effectively reduce blind spots in an LM. Retaining the model using WLDV is always recommended, as it reduced the number of UUs in every setting. If you aim to reduce the number of UUs with $c \geq 0.9$, we recommend generating synthetic samples using BLISS. If you aim to reduce the number of UUs with $c \geq 0.65$, we recommend using IBS. We want to emphasize that it was important to carefully determine the UU threshold τ. We recommend, in line with the definition of a UU by Attenberg et al. (Attenberg, Ipeirotis, and Provost 2015), not having a fixed threshold. Instead, we suggest that the confidence distribution should be taken into account during the evaluation process.

Being limited to text classification, this study lacks other modalities, such as image, voice, or video classification. A natural progression of this work is, therefore, to expand it into different modalities, such as image, video, and voice. Another promising direction for future research is the development of a standardized metric to assess the mitigation of blind spots. A metric should take into account both confidence and accuracy as components.

References

Almeida, Tiago and Jos Hidalgo (2012). *SMS Spam Collection*. Published: UCI Machine Learning Repository.

Attenberg, Joshua, Panos Ipeirotis, and Foster Provost (Mar. 2015). "Beat the Machine: Challenging Humans to Find a Predictive Model's "Unknown Unknowns"". en. In: *Journal of Data and Information Quality* 6.1, pp. 1–17. ISSN: 1936-1955, 1936-1963. DOI: 10.1145/2700832. URL: https://dl.acm.org/doi/10.1145/2700832 (visited on 07/05/2023).

Bansal, Gagan and Daniel Weld (Apr. 2018). "A Coverage-Based Utility Model for Identifying Unknown Unknowns". en. In: *Proceedings of the AAAI Conference on Artificial Intelligence* 32.1. ISSN: 2374-3468, 2159-5399. DOI: 10.1609/aaai.v32i1.11493. URL: https://ojs.aaai.org/index.php/AAAI/article/view/11493 (visited on 08/24/2023).

Cabrera, Ángel Alexander et al. (Oct. 2021). "Discovering and Validating AI Errors With Crowdsourced Failure Reports". en. In: *Proceedings of the ACM on Human-Computer Interaction* 5.CSCW2, pp. 1–22. ISSN: 2573-0142. DOI: 10.1145/3479569. URL: https://dl.acm.org/doi/10.1145/3479569 (visited on 07/05/2023).

Chawla, Nitesh V. et al. (2002). "SMOTE: synthetic minority over-sampling technique". In: *Journal of artificial intelligence research* 16. ISBN: 1076-9757, pp. 321–357.

Chung, Yeounoh et al. (Oct. 2019). *Unknown Examples & Machine Learning Model Generalization*. en. arXiv:1808.08294 [cs, stat]. URL: http://arxiv.org/abs/1808.08294 (visited on 07/05/2023).

Cook, R. Dennis (Feb. 1977). "Detection of Influential Observation in Linear Regression". In: *Technometrics* 19.1. Publisher: Taylor & Francis _eprint: https://doi.org/10.1080/00401706.1977.10489493, pp. 15–18. ISSN: 0040-1706. DOI: 10.1080/00401706.1977.10489493. URL: https://doi.org/10.1080/00401706.1977.10489493 (visited on 01/29/2024).

Devlin, Jacob et al. (May 2019). *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. en. arXiv:1810.04805 [cs]. URL: http://arxiv.org/abs/1810.04805 (visited on 05/15/2023).

Dixit, Tanay et al. (Nov. 2022). *CORE: A Retrieve-then-Edit Framework for Counterfactual Data Generation*. en. arXiv:2210.04873 [cs]. URL: http://arxiv.org/abs/2210.04873 (visited on 07/12/2023).

Dolan, Bill and Chris Brockett (2005). "Automatically constructing a corpus of sentential paraphrases". In: *Third International Workshop on Paraphrasing (IWP2005)*. URL: https://www.microsoft.com/en-us/research/publication/automatically-constructing-a-corpus-of-sentential-paraphrases/ (visited on 01/07/2024).

Finn, Chelsea, Pieter Abbeel, and Sergey Levine (July 2017). "Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks". en. In: *Proceedings of the 34th International Conference on Machine Learning*. ISSN: 2640-3498. PMLR, pp. 1126–1135. URL: https://proceedings.mlr.press/v70/finn17a.html (visited on 01/29/2024).

Gao, Ji et al. (May 2018). *Black-box Generation of Adversarial Text Sequences to Evade Deep Learning Classifiers*. en. arXiv:1801.04354 [cs]. URL: http://arxiv.org/abs/1801.04354 (visited on 07/11/2023).

Ghorbani, Amirata, Michael P. Kim, and James Zou (Feb. 2020). *A Distributional Framework for Data Valuation*. arXiv:2002.12334 [cs, stat]. DOI: 10.48550/arXiv.2002.12334. URL: http://arxiv.org/abs/2002.12334 (visited on 10/19/2023).

Ghorbani, Amirata and James Zou (May 2019). "Data Shapley: Equitable Valuation of Data for Machine Learning". en. In: *Proceedings of the 36th International Conference on Machine Learning*. ISSN: 2640-3498. PMLR, pp. 2242–2251. URL: https://proceedings.mlr.press/v97/ghorbani19c.html (visited on 01/29/2024).

Guu, Kelvin et al. (Nov. 2020). "Retrieval Augmented Language Model Pre-Training". en. In: *Proceedings of the 37th International Conference on Machine Learning*. ISSN: 2640-3498. PMLR, pp. 3929–3938. URL: https://proceedings.mlr.press/v119/guu20a.html (visited on 08/18/2023).

Han, Dongge et al. (Oct. 2023). "Replication Robust Payoff Allocation in Submodular Cooperative Games". In: *IEEE Transactions on Artificial Intelligence* 4.5. Conference Name: IEEE Transactions on Artificial Intelligence, pp. 1114–1128. ISSN: 2691-4581. DOI: 10.1109/TAI.2022.3195686. URL: https://ieeexplore.ieee.org/abstract/document/9847089?casa_token=C0u_Biq9wU8AAAAA:7cysUtKTpAZe26OHnFLDAyCxisjEsXpO68egWgy8ShSEDHs-t7iFfLk0NzCVFtZbHdMFXFRq (visited on 01/29/2024).

Han, Lei, Xiao Dong, and Gianluca Demartini (Oct. 2021). "Iterative Human-in-the-Loop Discovery of Unknown Unknowns in Image Datasets". en. In: *Proceedings of the AAAI Conference on Human Computation and Crowdsourcing* 9, pp. 72–83. ISSN: 2769-1349. DOI: 10.1609/hcomp.v9i1.18941. URL: https://ojs.aaai.org/index.php/HCOMP/article/view/18941 (visited on 01/26/2024).

Hashimoto, Tatsunori (July 2021). "Model Performance Scaling with Multiple Data Sources". In: *Proceedings of the 38th International Conference on Machine Learning*. Ed. by Marina Meila and Tong Zhang. Vol. 139. Proceedings of Machine Learning Research. PMLR, pp. 4107–4116. URL: https://proceedings.mlr.press/v139/hashimoto21a.html.

He, Tianxing et al. (May 2023). *On the Blind Spots of Model-Based Evaluation Metrics for Text Generation*. en. arXiv:2212.10020 [cs]. URL: http://arxiv.org/abs/2212.10020 (visited on 07/11/2023).

IEEE Symposium on Security and Privacy (2018). *Black-box Generation of Adversarial Text Sequences to Evade Deep Learning Classifiers*. URL: https://www.youtube.com/watch?v=Ho3V_eACoSQ (visited on 01/29/2024).

Jiang, Kevin Fu et al. (Oct. 2023). *OpenDataVal: a Unified Benchmark for Data Valuation*. en. arXiv:2306.10577 [cs, stat]. URL: http://arxiv.org/abs/2306.10577 (visited on 10/25/2023).

Jin, Di et al. (Apr. 2020). *Is BERT Really Robust? A Strong Baseline for Natural Language Attack on Text Classification and Entailment*. en. arXiv:1907.11932 [cs]. URL: http://arxiv.org/abs/1907.11932 (visited on 07/11/2023).

Kang, Jiawen, Zehui Xiong, Dusit Niyato, Shengli Xie, et al. (2019). *Incentive Mechanism for Reliable Federated Learning: A Joint Optimization Approach to Combining Reputation and Contract Theory*. Tech. rep.

Kang, Jiawen, Zehui Xiong, Dusit Niyato, Han Yu, et al. (Aug. 2019). "Incentive Design for Efficient Federated Learning in Mobile Networks: A Contract Theory Approach". In: *2019 IEEE VTS Asia Pacific Wireless Communications Symposium (APWCS)*, pp. 1–5. DOI: 10.1109/VTS-APWCS.2019.8851649. URL: https://ieeexplore.ieee.org/abstract/document/8851649?casa_token=RD-sd3LdOt0AAAAA:5YfvHvOUSii0j0j9yNATycg5UIbfIpVPt9nn3kFjEQU6jw6gO9t8KxidSWmzN7TrQ0oqhok0 (visited on 01/29/2024).

Lakkaraju, Himabindu et al. (Feb. 2017). "Identifying Unknown Unknowns in the Open World: Representations and Policies for Guided Exploration". en. In: *Proceedings of the AAAI Conference on Artificial Intelligence* 31.1. ISSN: 2374-3468, 2159-5399. DOI: 10.1609/aaai.v31i1.10821. URL: https://ojs.aaai.org/index.php/AAAI/article/view/10821 (visited on 07/10/2023).

Lehrer, E. (June 1988). "An axiomatization of the Banzhaf value". en. In: *International Journal of Game Theory* 17.2, pp. 89–99. ISSN: 1432-1270. DOI: 10.1007/BF01254541. URL: https://doi.org/10.1007/BF01254541 (visited on 01/29/2024).

Li, Qinbin et al. (Apr. 2023). "A Survey on Federated Learning Systems: Vision, Hype and Reality for Data Privacy and Protection". In: *IEEE Transactions on Knowledge and Data Engineering* 35.4. Conference Name: IEEE Transactions on Knowledge and Data Engineering, pp. 3347–3366. ISSN: 1558-2191. DOI: 10.1109/TKDE.2021.3124599. URL: https://ieeexplore.ieee.org/

abstract/document/9599369?casa_token=WOpVaEmzEWUAAAAA:D6K-MrHEiM26m8IV49jLKEqQDWFLfJdEaYVBTnUNQP_nK7q2prZnXJp_b2jH1rejWiPahgwl (visited on 01/29/2024).

Lippmann, Philip, Matthijs Spaan, and Jie Yang (Mar. 2024). *Exploring LLMs as a Source of Targeted Synthetic Textual Data to Minimize High Confidence Misclassifications*. en. arXiv:2403.17860 [cs]. DOI: 10.48550/arXiv.2403.17860. URL: http://arxiv.org/abs/2403.17860 (visited on 03/27/2024).

Lundberg, Scott and Su-In Lee (Nov. 2017). *A Unified Approach to Interpreting Model Predictions*. en. arXiv:1705.07874 [cs, stat]. URL: http://arxiv.org/abs/1705.07874 (visited on 03/11/2024).

Maas, Andrew L. et al. (June 2011). "Learning Word Vectors for Sentiment Analysis". In: *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*. Ed. by Dekang Lin, Yuji Matsumoto, and Rada Mihalcea. Portland, Oregon, USA: Association for Computational Linguistics, pp. 142–150. URL: https://aclanthology.org/P11-1015 (visited on 01/07/2024).

Morris, John X., Jin Yong Yoo, and Yanjun Qi (Oct. 2020). *TextAttack: Lessons learned in designing Python frameworks for NLP*. en. arXiv:2010.01724 [cs]. URL: http://arxiv.org/abs/2010.01724 (visited on 06/06/2023).

Mrkšić, Nikola et al. (Mar. 2016). *Counter-fitting Word Vectors to Linguistic Constraints*. arXiv:1603.00892 [cs]. DOI: 10.48550/arXiv.1603.00892. URL: http://arxiv.org/abs/1603.00892 (visited on 01/29/2024).

Niculescu-Mizil, Alexandru and Rich Caruana (Aug. 2005). "Predicting good probabilities with supervised learning". In: *Proceedings of the 22nd international conference on Machine learning*. ICML '05. New York, NY, USA: Association for Computing Machinery, pp. 625–632. ISBN: 978-1-59593-180-1. DOI: 10.1145/1102351.1102430. URL: https://dl.acm.org/doi/10.1145/1102351.1102430 (visited on 02/06/2024).

Page, Lawrence et al. (1999). "The pagerank citation ranking: Bringing order to the web". In: Publisher: Citeseer.

Paranjape, Bhargavi, Matthew Lamm, and Ian Tenney (2022). "Retrieval-guided Counterfactual Generation for QA". en. In: *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Dublin, Ireland: Association for Computational Linguistics, pp. 1670–1686. DOI: 10.18653/v1/2022.acl-long.117. URL: https://aclanthology.org/2022.acl-long.117 (visited on 07/13/2023).

Perez, Ethan et al. (Feb. 2022). *Red Teaming Language Models with Language Models*. en. arXiv:2202.03286 [cs]. URL: http://arxiv.org/abs/2202.03286 (visited on 07/14/2023).

Qi, Minfeng et al. (Sept. 2022). "A Hybrid Incentive Mechanism for Decentralized Federated Learning". In: *Distributed Ledger Technologies: Research and Practice* 1.1, 4:1–4:15. DOI: 10.1145/3538226. URL: https://dl.acm.org/doi/10.1145/3538226 (visited on 01/29/2024).

Rajpurkar, Pranav et al. (Oct. 2016). *SQuAD: 100,000+ Questions for Machine Comprehension of Text*. en. arXiv:1606.05250 [cs]. URL: http://arxiv.org/abs/1606.05250 (visited on 01/07/2024).

Roelofs, Rebecca et al. (Mar. 2022). "Mitigating Bias in Calibration Error Estimation". In: *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics*. Ed. by Gustau Camps-Valls, Francisco J. R. Ruiz, and Isabel Valera. Vol. 151. Proceedings of Machine Learning Research. PMLR, pp. 4036–4054. URL: https://proceedings.mlr.press/v151/roelofs22a.html.

Roth, Alvin E. (Oct. 1988). *The Shapley value: essays in honor of Lloyd S. Shapley*. en. Google-Books-ID: JK7MKu2A9cIC. Cambridge University Press. ISBN: 978-0-521-36177-4.

Sanh, Victor et al. (Feb. 2020). *DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter*. arXiv:1910.01108 [cs]. DOI: 10.48550/arXiv.1910.01108. URL: http://arxiv.org/abs/1910.01108 (visited on 01/24/2024).

Shi, Yuxin, Han Yu, and Cyril Leung (2023). “Towards Fairness-Aware Federated Learning”. In: *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–17. DOI: 10.1109/TNNLS.2023.3263594.

Silva Filho, Telmo et al. (Sept. 2023). “Classifier calibration: a survey on how to assess and improve predicted class probabilities”. en. In: *Machine Learning* 112.9, pp. 3211–3260. ISSN: 0885-6125, 1573-0565. DOI: 10.1007/s10994-023-06336-7. URL: https://link.springer.com/10.1007/s10994-023-06336-7 (visited on 02/22/2024).

Silverman, Bernard W. (Apr. 1986). *Density Estimation for Statistics and Data Analysis*. en. CRC Press. ISBN: 978-0-412-24620-3.

Sim, Rachael Hwee Ling, Xinyi Xu, and Bryan Kian Hsiang Low (July 2022). “Data Valuation in Machine Learning: ”Ingredients”, Strategies, and Open Challenges”. en. In: *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence*. Vienna, Austria: International Joint Conferences on Artificial Intelligence Organization, pp. 5607–5614. ISBN: 978-1-956792-00-3. DOI: 10.24963/ijcai.2022/782. URL: https://www.ijcai.org/proceedings/2022/782 (visited on 10/02/2023).

Sim, Rachael Hwee Ling, Yehong Zhang, et al. (July 2020). “Collaborative Machine Learning with Incentive-Aware Model Rewards”. In: *Proceedings of the 37th International Conference on Machine Learning*. Ed. by Hal Daumé III and Aarti Singh. Vol. 119. Proceedings of Machine Learning Research. PMLR, pp. 8927–8936. URL: https://proceedings.mlr.press/v119/sim20a.html.

Vandenhof, Colin (Oct. 2019). “A Hybrid Approach to Identifying Unknown Unknowns of Predictive Models”. en. In: *Proceedings of the AAAI Conference on Human Computation and Crowdsourcing* 7, pp. 180–187. ISSN: 2769-1349, 2769-1330. DOI: 10.1609/hcomp.v7i1.5274. URL: https://ojs.aaai.org/index.php/HCOMP/article/view/5274 (visited on 08/29/2023).

Vilalta, Ricardo and Youssef Drissi (June 2002). “A Perspective View and Survey of Meta-Learning”. en. In: *Artificial Intelligence Review* 18.2, pp. 77–95. ISSN: 1573-7462. DOI: 10.1023/A:1019956318069. URL: https://doi.org/10.1023/A:1019956318069 (visited on 01/29/2024).

Wang, Alex et al. (2018). “GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding”. en. In: *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*. Brussels, Belgium: Association for Computational Linguistics, pp. 353–355. DOI: 10.18653/v1/W18-5446. URL: http://aclweb.org/anthology/W18-5446 (visited on 01/07/2024).

Wiechmann, Marcel (Dec. 2023). *Enron Spam Dataset*. original-date: 2021-05-04T13:54:13Z. URL: https://github.com/MWiechmann/enron_spam_data (visited on 01/07/2024).

Wu, Zhaoxuan, Yao Shu, and Bryan Kian Hsiang Low (June 2022). “DAVINZ: Data Valuation using Deep Neural Networks at Initialization”. en. In: *Proceedings of the 39th International Conference on Machine Learning*. ISSN: 2640-3498. PMLR, pp. 24150–24176. URL: https://proceedings.mlr.press/v162/wu22j.html (visited on 01/29/2024).

Yan, Tom and Ariel D. Procaccia (May 2021). “If You Like Shapley Then You’ll Love the Core”. en. In: *Proceedings of the AAAI Conference on Artificial Intelligence* 35.6. Number: 6, pp. 5751–5759. ISSN: 2374-3468. DOI: 10.1609/aaai.v35i6.16721. URL: https://ojs.aaai.org/index.php/AAAI/article/view/16721 (visited on 01/29/2024).

Yoon, Jinsung, Sercan Arik, and Tomas Pfister (2020). “Data valuation using reinforcement learning”. In: *International Conference on Machine Learning*. PMLR, pp. 10842–10851. URL: http://proceedings.mlr.press/v119/yoon20a.html (visited on 10/02/2023).

Zhang, Jingwen, Yuezhou Wu, and Rong Pan (June 2021). “Incentive Mechanism for Horizontal Federated Learning Based on Reputation and Reverse Auction”. In: *Proceedings of the Web Conference 2021*. WWW ’21. New York, NY, USA: Association for Computing Machinery, pp. 947–956. ISBN: 978-1-

4503-8312-7. DOI: 10.1145/3442381.3449888. URL: https://dl.acm.org/doi/10.1145/3442381.3449888 (visited on 01/29/2024).

Prompts

The prompts to generate the synthetic samples were directly taken from the study of Lippmann et al. (Lippmann, Spaan, and Yang 2024), and can be found in their appendix B.