Uncertainty Estimation on Natural Language Processing

Jianfeng He

Dissertation submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy
in
Computer Science and Applications

Chang-Tien Lu, Chair
Chandan K. Reddy
Jin-Hee Cho
Dawei Zhou
Bei Xiao

April 25, 2024
Falls Church, Virginia

Keywords: Uncertainty Estimation, Bayesian Neural Network, Evidential Neural Network, Text Classification, Few-Shot, Named Entity Recognition, Text Summarization.

Uncertainty Estimation on Natural Language Processing

Jianfeng He

(ABSTRACT)

Text plays a pivotal role in our daily lives, encompassing various forms such as social media posts, news articles, books, reports, and more. Consequently, Natural Language Processing (NLP) has garnered widespread attention. This technology empowers us to undertake tasks like text classification, entity recognition, and even crafting responses within a dialogue context. However, despite the expansive utility of NLP, it frequently necessitates a critical decision: whether to place trust in a model's predictions. To illustrate, consider a state-of-the-art (SOTA) model entrusted with diagnosing a disease or assessing the veracity of a rumor. An incorrect prediction in such scenarios can have dire consequences, impacting individuals' health or tarnishing their reputation. Consequently, it becomes imperative to establish a reliable method for evaluating the reliability of an NLP model's predictions, which is our focus-uncertainty estimation on NLP.

Though many works have researched uncertainty estimation or NLP, the combination of these two domains is rare. This is because most NLP research emphasizes model prediction performance but tends to overlook the reliability of NLP model predictions. Additionally, current uncertainty estimation models may not be suitable for NLP due to the unique characteristics of NLP tasks, such as the need for more fine-grained information in named entity recognition. Therefore, this dissertation proposes novel uncertainty estimation methods for different NLP tasks by considering the NLP task's distinct characteristics.

The NLP tasks are categorized into natural language understanding (NLU) and natural language generation (NLG, such as text summarization). Among the NLU tasks, the understanding could be on two views, global-view (e.g. text classification at document level) and local-view (e.g. natural language inference at sentence level and named entity recognition at token level). As a result, we research uncertainty estimation on three tasks: text classification, named entity recognition, and text summarization. Besides, because few-shot text classification has captured much attention recently, we also research the uncertainty estimation on few-shot text classification.

For the first topic, uncertainty estimation on text classification, few uncertainty models focus on improving the performance of text classification where human resources are involved. In response to this gap, our research focuses on enhancing the accuracy of uncertainty scores by bolstering the confidence associated with winning scores. we introduce MSD, a novel model comprising three distinct components: "mix-up," "self-ensembling," and "distinctiveness score." The primary objective of MSD is to refine the accuracy of uncertainty scores by mitigating the issue of overconfidence in winning scores while simultaneously considering various categories of uncertainty. seamlessly integrate with different Deep Neural Networks. Extensive experiments with ablation settings are conducted on four real-world datasets, resulting in consistently competitive improvements.

Our second topic focuses on uncertainty estimation on few-shot text classification (UEFTC), which has few or even only one available support sample for each class. UEFTC represents an underexplored research domain where, due to limited data samples, a UEFTC model predicts an uncertainty score to assess the likelihood of classification errors. However, traditional uncertainty estimation models in text classification are ill-suited for UEFTC since they demand extensive training data, while UEFTC operates in a few-shot scenario, typically providing just a few support samples, or even just one, per class. To tackle this challenge, we introduce Contrastive Learning from Uncertainty Relations (CLUR) as a solution tailored for UEFTC. CLUR exhibits the unique capability to be effectively trained with only one support sample per class, aided by pseudo uncertainty scores. A distinguishing feature of CLUR is its autonomous learning of these pseudo uncertainty scores, in contrast to previous approaches that relied on manual specification. Our investigation of CLUR encompasses four model structures, allowing us to evaluate the performance of three commonly employed contrastive learning components in the context of UEFTC. Our findings highlight the effectiveness of two of these components.

Our third topic focuses on uncertainty estimation on sequential labeling. Sequential labeling involves the task of assigning labels to individual tokens in a sequence, exemplified by Named Entity Recognition (NER). Despite significant advancements in enhancing NER performance in prior research, the realm of uncertainty estimation for NER (UE-NER) remains relatively uncharted but is of paramount importance. This topic focuses on UE-NER, seeking to gauge uncertainty scores for NER predictions. Previous models for uncertainty estimation often overlook two distinctive attributes of NER: the interrelation among entities (where the learning of one entity's embedding depends on others) and the challenges posed by incorrect span predictions in entity extraction. To address these issues, we introduce the Sequential Labeling Posterior Network (SLPN), designed to estimate uncertainty scores for the extracted entities while considering uncertainty propagation from other tokens. Additionally, we have devised an evaluation methodology tailored to the specific nuances of wrong-span cases.

Our fourth topic focuses on an overlooked question that persists regarding the evaluation reliability of uncertainty estimation in text summarization (UE-TS). Text summarization, a key task in natural language generation (NLG), holds significant importance, particularly in domains where inaccuracies can have serious consequences, such as healthcare. UE-TS has garnered attention due to the potential risks associated with erroneous summaries. However, the reliability of evaluating UE-TS methods raises concerns, stemming from the interdependence between uncertainty model metrics and the wide array of NLG metrics. To address these concerns, we introduce a comprehensive UE-TS benchmark incorporating twenty-six NLG metrics across four dimensions. This benchmark evaluates the uncertainty estimation capabilities of two large language models and one pre-trained language model across two datasets. Additionally, it assesses the effectiveness of fourteen common uncertainty estimation methods. Our study underscores the necessity of utilizing diverse, uncorrelated NLG metrics and uncertainty estimation techniques for a robust evaluation of UE-TS methods.

Uncertainty Estimation on Natural Language Processing

Jianfeng He

(GENERAL AUDIENCE ABSTRACT)

Text is integral to our daily activities, appearing in various forms such as social media posts, news articles, books, and reports. We rely on text for communication, information dissemination, and decision-making. Given its ubiquity, the ability to process and understand text through Natural Language Processing (NLP) has become increasingly important. NLP technology enables us to perform tasks like text classification, which involves categorizing text into predefined labels, and named entity recognition (NER), which identifies specific entities such as names, dates, and locations within text. Additionally, NLP facilitates generating coherent and contextually appropriate responses in conversational agents, enhancing human-computer interaction. However, the reliability of NLP models is crucial, especially in sensitive applications like medical diagnoses, where errors can have severe consequences.

This dissertation focuses on uncertainty estimation in NLP, a less explored but essential area. Uncertainty estimation helps evaluate the confidence of NLP model predictions. We propose new methods tailored to various NLP tasks, acknowledging their unique needs.

NLP tasks are divided into natural language understanding (NLU) and natural language generation (NLG). Within NLU, we look at tasks from two perspectives: a global view (e.g., document-level text classification) and a local view (e.g., sentence-level inference and token-level entity recognition). Our research spans text classification, named entity recognition (NER), and text summarization, with a special focus on few-shot text classification due to its recent prominence.

For text classification, we introduce the MSD model, which includes three components to enhance uncertainty score accuracy and address overconfidence issues. This model integrates seamlessly with different neural networks and shows consistent improvements in experiments.

For few-shot text classification, we develop Contrastive Learning from Uncertainty Relations (CLUR), designed to work effectively with minimal support samples per class. CLUR autonomously learns pseudo uncertainty scores, demonstrating effectiveness with various contrastive learning components.

In NER, we address the unique challenges of entity interrelation and span prediction errors. We propose the Sequential Labeling Posterior Network (SLPN) to estimate uncertainty scores while considering uncertainty propagation from other tokens.

For text summarization, we create a benchmark with tens of metrics to evaluate uncertainty estimation methods across two datasets. This benchmark helps assess the reliability of these methods, highlighting the need for diverse, uncorrelated metrics.

Overall, our work advances the understanding and implementation of uncertainty estimation in NLP, providing more reliable and accurate predictions across different tasks.

Acknowledgments

I am profoundly grateful to my advisor, Professor Chang-Tien Lu, for his unwavering support throughout my Ph.D. journey. His guidance and profound insights into research have been invaluable to me. Beyond academic matters, he has been a mentor and a source of guidance in both my career and personal life. I am deeply appreciative of his support and encouragement during the most challenging moments of my Ph.D. years. Working with him has been an absolute privilege.

I extend my sincere thanks to my Ph.D. committee members, Professors Jin-Hee Cho, Dawei Zhou, Chander K. Reddy, and Bei Xiao, for their support and invaluable suggestions.

Special appreciation goes to Professors Feng Chen and Ming Jin, as well as Dr. Xuchao Zhang and Dr. Shuo Lei, for their continuous support and assistance. Their exceptional dedication to research and inspiring personalities have been a constant source of motivation for me.

I am also grateful to my friends in the Spatial Data Management Laboratory, including Shuo Lei, Lei Zhang, Fanglan Chen, Xuchao Zhang, Zhiqian Chen, Kaiqun Fu, Taoran Ji, Ting Hua, Abdulaziz Alhamadani, Lulwah AlKulaib, Shailik Sarkar, Yanshen Sun, Yuer Jiang, Linhan Wang, Shengkun Wang, and Min Zhang, among others. Their companionship and support have made this journey memorable and enjoyable.

Also, sincere thanks to Prof. Rebecca Hwa at the University of Pittsburgh, who inspired me on how to conduct research at the very beginning of my Ph.D. study.

Further, I greatly appreciate the instruction of my master's advisors. They are Prof. Bingpeng Ma, Prof. Shuhui Wang, and Prof. Qingming Huang.

I deeply appreciate the discussions and interactions with my friends at The University of Texas at Dallas, including Linlin Yu, Changbin Li, and Kangshuo Li.

Plus, great thanks to other classmates. They are Dr. Yang Xiao, Dr. Ning Wang, Dr. Ruide Zhang, Shanghao Shi, Hexuan Yu, and Chaoyu Zhang.

My heartfelt thanks also go to my roommates, Zheyu Zhang, Dan Zhao, Shaoyu Li, Heng Jin, Yi Wei, Hang Zhao, Kang Shen, Yimin Chen, Di Zhang, and Yichuan Li, for their unwavering support over the past five years.

Lastly, I dedicate this dissertation to my parents and family, whose endless love and support have been my guiding light throughout this journey. They are the reason behind my perseverance and success.

Contents

[1]J. He, X. Zhang, S. Lei, Z. Chen, F. Chen, A. Alhamadani, B. Xiao and C. Lu. Towards More Accurate Uncertainty Estimation In Text Classification [C]. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: 8362-8372.

[2]J.He, X. Zhang, S. Lei, F. Chen, A. Alhamadani, B. Xiao, C. Lu. CLUR: Uncertainty Estimation for Few-Shot Text Classification with Contrastive Learning. [C] In Proceedings of the 29th ACM SIGKDD international conference on knowledge discovery & data mining.

[3] J. He, Y. Lin, S. Lei, C. Lu, F. Chen. Uncertainty Estimation on Sequential Labeling via Uncertainty Transmission. ArXiv preprint arXiv:2311.08726 (2023).

[4] J.He, M. Jin, L. Yu, C. Li, R. Yang, R. Jia, F. Chen, C. Lu. Can We Trust the Performance Evaluation of Uncertainty Estimation Methods in Text Summarization? Will be submitted to NIPS 2024.

List of Figures

List of Tables

Chapter 1

Introduction

Text plays a pivotal role in our daily lives, permeating various mediums such as social media, news articles, books, and reports. This ubiquity has propelled Natural Language Processing (NLP) into the limelight, attracting widespread attention. NLP technology offers a versatile toolkit, enabling tasks ranging from text topic classification, entity recognition, to generating responses within a dialogue context. However, despite the broad spectrum of NLP applications, it consistently confronts a critical challenge: the determination of whether to place confidence in a model's predictions. Consider, for instance, a state-of-the-art (SOTA) model entrusted with diagnosing a novel disease or assessing the veracity of a rumor. An erroneous prediction in such instances carries severe repercussions, potentially jeopardizing the health and reputation of individuals. Hence, there arises an imperative need for a reliable method to ascertain the reliability of NLP model predictions.

While a substantial body of research has delved into uncertainty estimation and Natural Language Processing (NLP) individually, the fusion of these two domains remains relatively scarce. This scarcity can be attributed to the fact that many NLP researchers prioritize model prediction performance while often overlooking the assessment of the reliability of those predictions. Furthermore, the existing uncertainty estimation models primarily cater to sample-level classification tasks, such as image classification. In contrast, besides document-level understanding, NLP tasks often demand a more nuanced understanding at finer-grained levels, encompassing activities like named entity recognition (NER) at the phrase level and natural language inference at the sentence level. Beyond understanding tasks, NLP also involves generation tasks, such as text summarization and dialogue response generation. Given the unique characteristics of NLP, it becomes apparent that conventional uncertainty estimation methods may not be directly applicable. Adaptations or entirely novel approaches need to be designed, taking into account NLP's characteristics. Consequently, it is important and essential to research uncertainty estimation on NLP.

NLP tasks can be broadly categorized into two domains: natural language understanding (NLU), such as text classification, and natural language generation (NLG), which includes

tasks like text summarization. Within NLU, understanding can take two perspectives: a global view, which pertains to tasks such as document-level text classification, and a local view, encompassing tasks like natural language inference at the sentence level and named entity recognition at the token level. Consequently, our research focuses on uncertainty estimation within three key tasks: text classification, named entity recognition, and text summarization. Furthermore, given the recent surge in interest surrounding few-shot text classification, we also delve into uncertainty estimation on few-shot text classification. We clarify our research tasks and their applications as below.

- **Uncertainty Estimation on Text Classification.** Text classification stands as a widely explored subject, boasting a wide array of practical applications. A prevailing and widely used model for text classification is the Deep Neural Network (DNN). However, certain real-world scenarios demand outcomes of even greater precision than those achieved by cutting-edge algorithms, such as rumor or emergency news detection. Consequently, the most uncertain predictions necessitate the input of domain experts for further deliberation [189]. To make efficient use of limited human resources, it becomes crucial to compute a measure of uncertainty associated with the model's predictions. This measure quantifies the level of confidence (or lack thereof) in the model's predictions. The primary goal of this work is to enhance the accuracy of uncertainty scores using DNNs in the context of text classification while involving human expertise in the testing phase. It is worth noting that this approach differs from active learning, which primarily entails expert participation during the training phase.

- **Uncertainty Estimation on Few-Shot Text Classification.** Few-shot text classification involves training a classifier using a limited set of training texts, as discussed in previous research. In such few-shot scenarios, a critical decision often arises regarding whether to place trust in the model's outputs. For instance, consider a state-of-the-art (SOTA) model tasked with diagnosing a new disease; it begins with access to only a few descriptions of the condition but requires high accuracy. To enhance classification accuracy, one approach is to have experts review the most uncertain results. However, experts are both expensive and in short supply. Consequently, uncertainty estimation plays a pivotal role in optimizing decision-making and conserving expert resources across various few-shot applications. This topic aims to enhance the accuracy of Uncertainty Estimation for Few-shot Text Classification (UEFTC). Specifically, UEFTC assesses the probability that a classification result is incorrect when dealing with a limited training dataset [1]. UEFTC models should assign high uncertainty scores to potentially misclassified predictions and low uncertainty scores to correct predictions.

- **Uncertainty Estimation on Sequential Labeling.** Named Entity Recognition (NER) is a fundamental task within the information extraction domain. This task involves two core steps: identifying spans of entities and assigning appropriate labels to these entities. The accuracy of NER predictions holds significant implications across

various information extraction scenarios. For instance, extracting inaccurate temporal information can lead to erroneous policy analysis, while misclassifying a person's name as a time entity can potentially result in a privacy breach. Hence, it becomes imperative to establish the reliability of NER predictions. Therefore, our primary objective is to advance the field of Uncertainty Estimation in Named Entity Recognition (UE-NER). UE-NER specifically focuses on developing techniques and methodologies to quantify the confidence and uncertainty associated with NER predictions. By enhancing our ability to assess the reliability of NER outputs, we aim to provide decision-makers and analysts with valuable models for making informed judgments and decisions in a wide range of applications and domains.

- **Uncertainty Estimation on Text Summarization.** Text summarization stands as a quintessential task within NLG. Particularly in risk-critical domains like healthcare, where misinterpretations can lead to biased diagnoses based on inaccurate symptom summaries, the need for uncertainty estimation in text summarization (UE-TS) has garnered significant attention from researchers. However, a lingering concern persists regarding the reliability of UE-TS performance evaluations. This concern stems from the reliance on uncertainty model metrics on NLG metrics, as well as the diversity inherent in NLG metric selection. To address this fundamental question and its corollaries, we introduce a comprehensive UE-TS benchmark integrating twenty-six NLG metrics across four dimensions. This benchmark rigorously evaluates the uncertainty estimation capabilities of two large language models and one pre-trained language model across two distinct datasets. Furthermore, within this benchmark framework, we scrutinize fourteen common uncertainty estimation methods to gauge their effectiveness. Our findings underscore the critical importance of incorporating multiple, uncorrelated NLG metrics and uncertainty estimation methods to ensure a robust and effective evaluation of uncertainty estimation techniques. By doing so, we pave the way for more reliable assessments and advancements in uncertainty estimation methods within the realm of text summarization.

1.1 Research Issues

1.1.1 Uncertianty Estimation on Text Classification

This research addresses the overconfidence in uncertainty estimation and consider uncertainty in a comprehensive way. Concretely, though various metrics of the uncertainty score have been studied [24, 79, 140, 161, 173], the existing metrics directly or indirectly depend on *winning score*, which is the maximum probability in a semantic vector (softmax vector from the last layer of a DNN model) [151]. Therefore, improving *Confidence of Winning Score* (CWS), which describes how confident the winning score matches the sample uncertainty and represents the accuracy of the winning score, is helpful to improve the accuracy of

uncertainty score. To show the effect of improving CWS, this paper considers a basic way to measure uncertainty score, which is the reciprocal of winning score [143]. However, we face two challenges in improving CWS.

- **Reduce the effect of overconfidence of winning score.** The prior research in NLP has largely overlooked the issue of overconfidence in predicting winning scores. We have identified the presence of overconfidence in our training samples. This overconfidence arises from the fact that all training samples have their winning scores set to 1 using one-hot labels, resulting in each sample having the same level of uncertainty. Consequently, the uncertainty in our training samples is uniform across the board. This uniformity means that the winning scores and sample uncertainty remain constant across various training samples, making it impossible to guarantee a negative correlation between winning scores and sample uncertainty. This negative correlation is an adverse consequence of overconfidence and has implications for how we calculate uncertainty scores. Specifically, during the testing phase, we employ different predicted winning scores to match varying sample uncertainty levels based on the assumption that predicted winning scores are negatively correlated with sample uncertainty. However, this assumption is biased due to the detrimental impact of overconfidence.
- **Generate winning scores by considering comprehensive categories of uncertainty in a comprehensive way**. Moreover, when generating the winning score, it's crucial to account for the influence of various categories of uncertainty simultaneously. In contrast, most of the previous studies [140, 161, 173, 189] have only focused on one or two categories of uncertainty at a time. We hypothesize that this limited consideration of uncertainty categories may lead to a decrease in the Corrected Winning Score (CWS) and consequently affect the accuracy of the uncertainty score. To validate this hypothesis, we conducted ablation experiments.

1.1.2 Uncertainty Estimation on Few-Shot Text Classification.

The few-shot setting in UEFTC makes many uncertainty estimation methods difficult to use. Concretely, compared to traditional uncertainty estimation tasks, the few-shot setting in UEFTC provides only a few support samples or even one support sample (1-shot) per class in each episode [1]. The issues in this task are listed as below

- **The current distribution-based methods in uncertainty estimation are hard to tackle UEFTC given the few-shot limitation.** Bayesian Neural Networks (BNN)-based methods learn a distribution over the model parameters [104, 125], or learn a distribution for each semantic class [9, 10, 145]. Due to the few-support-sample limitation in UEFTC, it is difficult to learn a distribution by a few or just one

[1]The term "support" is explained in Sec. 3.3.1

support sample per class. As a result, the parameter-distribution-based and sample-distribution-based methods are not suitable for addressing UEFTC. Therefore, we need to address UEFTC in other ways.

- **Manually setting pseudo uncertainty scores as the ground truth has a more negative impact on UEFTC.** While pseudo-label-based methods have gained popularity in recent years [11, 166], their current utilization in uncertainty estimation [58] has a drawback. This drawback involves the manual assignment of pseudo uncertainty scores as the reference ground truth. Specifically, in previous pseudo-label-based approaches, coefficients from techniques like mix-up (as described in Zhang et al., 2017) have been manually designated as pseudo uncertainty scores during the training of uncertainty models. However, this manual setting of pseudo uncertainty scores can be problematic because we lack the actual ground-truth uncertainty scores for training samples based on a model's structure. In the few-shot learning scenario, inaccurate pseudo uncertainty scores have a more significant impact on Uncertainty Estimation for Few-shot Text Classification (UEFTC) compared to tasks with larger datasets. This is because each individual sample carries more weight in UEFTC. In a one-shot learning setting, an inaccurate pseudo uncertainty score means that the sole support sample is assigned an incorrect ground truth uncertainty score, which results in noticeable training bias

1.1.3 Uncertainty Estimation on Sequential Labeling.

Different from text classification, sequential labeling is a task that focuses on the classification of each token. Also, based on the token classification, it needs to additionally extract spans, which could lead to wrong spans that have no ground truth labels at all. Thus, we summarize the issues of this task as below.

- **The current uncertainty estimation methods ignore the uncertainty transmission between tokens.** Especially, current uncertainty estimation methods can be classified into two main categories: parameter-distribution-based methods, such as Bayesian Neural Networks (BNN) [104, 125], which learns a distribution over the model parameters; and sample-distribution-based methods, which calculate uncertainty scores based on the distribution of training samples [10, 58, 129]. These methods primarily focus on image or text classification, where correlations between different images or texts are weak or limited. Consequently, they overlook the uncertainty transmission inherent in sequential labeling. Since sequential labeling plays a pivotal role in Natural Language Processing (NLP), with NER as a representative example, it is imperative for us to address UE-NER by considering uncertainty transmission.
- **NER task involves an additional step, entity extraction, besides entity classification, which could lead to difficulty in uncertainty estimation.** In contrast

to previous text classification tasks [113], which focus solely on sample classification, NER tasks require the additional task of extracting entity spans, such as locating "Barack Obama."However, entity span extraction may predict entities with wrong span (WS), such as predicting "Obama was" as an entity. These WS entities lack ground truth entity labels and evaluating uncertainty estimation requires ground truth labels, thus these entities cannot be used for evaluating uncertainty estimation. Therefore, we require an innovative approach to evaluate a UE-NER model that takes into account these WS entities.

1.1.4 Uncertainty Estimation on Text Summarization.

Unlike text classification and sequential labeling, text summarization is a generation task characterized by diverse ground-truth summaries. Consequently, text summarization entails a variety of NLG metrics. Thus, we outline the challenges of this task as follows.

- **An overlooked question persists regarding the reliability of the UE-TS evaluation framework.** This concern arises from two reasons. Specifically, the first reason is that evaluation metrics for uncertainty estimation in NLG tasks rely on the alignment between the sample ranks obtained from the uncertainty scores and the sample rank based on the respective NLP metric scores. The second reason pertains to the label diversity in NLG tasks, such as text summarization, which leads to the utilization of various NLG metrics. Previous evaluations of UE-TS models [26, 36, 52] have collectively employed one or two uncertainty estimation metrics, yet each of these metrics depends solely on a single type of NLG metric. As a result, we have this reliability concern.
- **How to choose the representative NLG metrics and representative uncertainty methods for the research.** Specifically, for NLG metrics, there are numerous choices, and similarly, there are many options for uncertainty estimation methods. It could be very time-consuming to compare every possible NLG metric with every possible uncertainty estimation method. Additionally, the conclusions drawn may be specific to each method and may not necessarily represent other methods, albeit with some similarities. Therefore, identifying representative NLG metrics and uncertainty estimation methods could help us complete our research more efficiently and yield more representative conclusions for each category of NLG metrics and each category of uncertainty methods.

1.2 Contributions

The major contributions of the research presented here can be stated as follows:

1.2.1 Uncertainty Estimation on Text Classification

The main contributions of this work are summarized as follows:

- **Reducing impact of overconfidence.** To reduce the impact of overconfidence in calculating uncertainty scores, we apply mix-up to generate new sample representations boosting the negative correlation between the winning scores and sample uncertainty.
- **Considering various uncertainty comprehensively.** We propose MSD with three components to handle the epistemic uncertainty, aleatoric uncertainty, and parametric uncertainty simultaneously, so that the uncertainty score is more accurate.
- **Designing flexibility of MSD.** MSD can be applied with different DNNs (CNN, RNN, and Transformer). Each component in MSD can be assembled with other components arbitrarily due to their independence.
- **Implementing extensive experiments.** We evaluated MSD by the improvement of text classification accuracy in simulating human involvement. The experiments of MSD with ablation setting on four datasets achieved competitive results, which demonstrated that MSD generates more accurate uncertainty scores.

1.2.2 Uncertainty Estimation on Few-Shot Text Classification

- We present a novel framework to **Improving uncertainty estimation by a few or just one support sample per class.** To our knowledge, we are the first to solve UEFTC under its few-support-sample limitation. Our proposed CLUR can be trained with one support sample per class in each episode because it takes advantage of ensemble and pseudo-label-based methods. Our solution in UEFTC can also motivate uncertainty estimation in other few-shot applications.
- **Proposing and using uncertainty relations to self-adaptively learn pseudo uncertainty scores as the ground truth uncertainty.** To address the issue of manually setting pseudo uncertainty scores, we generate augmented sample pairs to self-adaptively learn their pseudo uncertainty scores by our proposed uncertainty relations. Unlike current contrastive learning models that only have equal relations (i.e., having the same ($=$) or different ($\neq$) classes) between the augmented samples, our uncertainty relations include additional unequal relations, that are larger ($>$) or smaller ($<$) uncertainty relations among the augmented sample uncertainty.
- **Investigating the performance of the three common-used contrastive learning components in UEFTC.** As the first study to apply contrastive learning in UEFTC, we also design four model structures in CLUR to investigate the performance of three common-used contrastive learning components in UEFTC. We find that two of

them are effective in UEFTC, enabling us to optimize CLUR. Future UEFTC models can benefit from our findings.

- **Conducting extensive experiments and benchmarking the UEFTC.** We demonstrate that CLUR effectively outperforms six baselines on four datasets (20News, RCV1, Amazon, and HuffPost), including an improvement of 4.52% AUPR on an RCV1 dataset in a 5-way 1-shot setting. We release our code as UEFTC benchmark.
- **Implementing extensive experiments.** We evaluated our proposed method in three datasets. The experiments with ablation setting achieved competitive results, which demonstrated that our proposed method predicts more accurate uncertainty scores in the uncertainty estimation of NER tasks.

1.2.3 Uncertainty Estimation on Sequential Labeling

- **Firstly considering uncertainty transmission in UE-NER** Since each token embedding is influenced by other tokens within a given text, and token embedding directly affects the uncertainty of predicted entity labels, we propose a novel method to transmit uncertainty between tokens using a revised self-attention. To the best of our knowledge, we are the first to consider uncertainty transmission in UE-NER.
- **Firstly evaluating OOD and WS tasks.** Because of the existence of WS entities in the NER task, we have found that traditional evaluation methods for uncertainty estimation are inapplicable in UE-NER. Therefore, we propose a novel uncertainty estimation evaluation to evaluate both OOD and WS detection tasks.

1.2.4 Uncertainty Estimation on Text Summarization

- **First raising a valid reliability concern regarding UE-TS.** As far as we know, we are the first to bring attention to a legitimate issue regarding the evaluation of UE-TS methods. To address this concern, we suggest assessing UE-TS methods using various NLG metrics. These NLG metrics cover four essential dimensions in NLG evaluation, namely coherence, consistency, fluency, and relevance, as outlined in [197].
- **Releasing a UE-TS benchmark exploring the relationship between NLG metrics and uncertainty estimation methods.** We introduce a UE-TS benchmark to evaluate uncertainty estimation from diverse NLG metric viewpoints, marking what we believe is the first attempt of its kind. This benchmark evaluates two Large Language Models (LLMs) and one Pre-trained Language Model (PLM) on two datasets specifically targeting UE-TS. Within this benchmark, we integrate twenty-six NLG metrics and fourteen uncertainty estimation methods.

- **Uncovering the findings to benefit future research on UE-TS.** We have revealed compelling discoveries as detailed in Section 5.6. Given that text summarization serves as a prominent example of an NLG task, our insights from text summarization tasks could similarly inspire the development and examination of uncertainty estimation in other NLG tasks.

Chapter 2

Uncertainty Estimation on Text Classification[1]

The uncertainty measurement of classified results is especially important in areas requiring limited human resources for higher accuracy. For instance, data-driven algorithms diagnosing diseases need accurate uncertainty score to decide whether additional but limited quantity of experts are needed for rectification. However, few uncertainty models focus on improving the performance of text classification where human resources are involved. To achieve this, we aim at generating accurate uncertainty score by improving the confidence of winning scores. Thus, a model called MSD, which includes three independent components as "mix-up", "self-ensembling", "distinctiveness score", is proposed to improve the accuracy of uncertainty score by reducing the effect of overconfidence of winning score and considering the impact of different categories of uncertainty simultaneously. MSD can be applied with different Deep Neural Networks. Extensive experiments with ablation setting are conducted on four real-world datasets, on which, competitive results are obtained.

2.1 Introduction

Text classification is a popular topic with broad applications. A successful and common model for text classification is Deep Neural Network (DNN). However, some real-world applications expect results with higher accuracy than the ones achieved by state-of-the-art algorithms. Hence, the most uncertain predictions need domain experts for further decisions [189]. To efficiently leverage the limited human resources, it is essential to calculate *uncertainty score* of the model prediction, which quantifies how unconfident the model pre-

[1]J. He, X. Zhang, S. Lei, Z. Chen, F. Chen, A. Alhamadani, B. Xiao and C. Lu. Towards More Accurate Uncertainty Estimation In Text Classification [C]. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: 8362-8372.

diction is. This paper aims at generating more accurate uncertainty score through DNNs in the text classification with human involvement in the testing process. This is different from active learning, which involves experts in the training process.

Though various metrics of the uncertainty score have been studied [24, 79, 140, 161, 173], the existing metrics directly or indirectly depend on *winning score*, which is the maximum probability in a semantic vector (softmax vector from the last layer of a DNN model) [151]. Therefore, improving *Confidence of Winning Score* (CWS), which describes how confident the winning score matches the sample uncertainty and represents the accuracy of the winning score, is helpful to improve the accuracy of uncertainty score. To show the effect of improving CWS, this paper considers a basic way to measure uncertainty score, which is the reciprocal of winning score [143]. However, we face two challenges in improving CWS: (1) how to reduce effect of *overconfidence of winning score*[2] to boost negative correlation between the winning score and sample uncertainty, (2) how to generate winning scores by considering comprehensive categories of uncertainty in one model rather than only one or two categories of uncertainty at a time.

The overconfidence of winning scores has been neglected by vast previous works in Natural Language Processing. We identify the presence of *overconfidence* for the training samples: because the winning scores of training samples are all set as 1 by one-hot labels, each sample will have the same uncertainty score. Consequently, the training sample uncertainty will be the same. Together, the winning scores and sample uncertainty are the same for various training samples. Hence, the negative correlation between the winning scores and sample uncertainty cannot be guaranteed, which is a negative effect of the overconfidence. The effect will affect calculating the uncertainty scores. Concretely, in the testing process, we apply different predicted winning scores to match different sample uncertainty based on a latent assumption that the predicted winning score is negatively correlated to the sample uncertainty. However, the assumption is biased because of the negative effect of the overconfidence. To mitigate the impact of overconfidence, we generate new training sample representations with different winning scores, which are also negatively correlated to the sample uncertainty.

Additionally, the process generating the winning score should consider the impact of different categories of uncertainty simultaneously, while vast of the previous works [140, 161, 173, 189] only consider one or two categories of uncertainty at a time[3]. We assume the partial consideration will decrease the CWS, and so will the accuracy of uncertainty score. We verify this assumption by our ablation experiments. The uncertainty of a model prediction is derived from two parts: *data uncertainty* and *model uncertainty*. The data uncertainty [138] is further divided into two categories: *epistemic uncertainty* comes from lack of knowledge, such as only few training data or out-of-distribution testing data; *aleatoric uncertainty* is caused by noisy data in the generation of both training data and testing data. The model uncertainty [97] also has two categories: *parametric uncertainty* comes from different possibilities

[2]Noted as overconfidence in the paper.

[3]Please refer to our appendix for detailed comparisons

of parameter values in estimating model parameters under the current model structure and training data; *structural uncertainty* is uncertainty about whether the current model design (e.g., layers, loss functions) is reasonable or sufficient for the current task and training data. Since the solution of structural uncertainty requires extremely high computations, such as Neural Architecture Search (NAS) [174,198], we only reduce or scale the other three categories of uncertainty simultaneously to improve the CWS.

To address the above two challenges, we propose a model called MSD, which is named as the initials of its components ("Mix-up", "Self-ensembling", and "Distinctiveness score") aiming at handling overconfidence and various uncertainty with flexibility. The flexibility means that MSD is effective on different DNN models (Convolutional Neural Network (CNN), Recurrent Neural Network (RNN) and Transformer [155]), and each component in MSD is independent, which can be arbitrarily assembled. The main contributions of our work can be summarized as follows,

Reducing impact of overconfidence. To reduce the impact of overconfidence in calculating uncertainty scores, we apply mix-up to generate new sample representations boosting the negative correlation between the winning scores and sample uncertainty.

Considering various uncertainty comprehensively. We propose MSD with three components to handle the epistemic uncertainty, aleatoric uncertainty, and parametric uncertainty simultaneously, so that the uncertainty score is more accurate.

Designing flexibility of MSD. MSD can be applied with different DNNs (CNN, RNN, and Transformer). Each component in MSD can be assembled with other components arbitrarily due to their independence.

Implementing extensive experiments. We evaluated MSD by the improvement of text classification accuracy in simulating human involvement. The experiments of MSD with ablation setting on four datasets achieved competitive results, which demonstrated that MSD generates more accurate uncertainty scores.

2.2 Related work

Methods mitigating uncertainty: One main solution to mitigate uncertainty is Bayesian Neural Network (BNN) [76], which is a neural network with a prior distribution on its weights. Based on BNN, variational Bayesian inference is proposed, which finds an approximated distribution of parameters for the true distribution of parameters by Kullback-Leibler (KL) divergence [103, 106, 168, 173]. Further, as an approximation of variational Bayesian inference, Monte Carlo dropout is proposed [30, 73]. This is implemented by training a model with dropout before every layer, and also performing the dropout in the testing process to derive results from different sampled parameter sets. Plus, an approximation of Monte Carlo dropout is tried by only adding dropout before the last layer [136, 143]. Besides BNN, noise injection is the other main technique to mitigate uncertainty. It has two categories: parameter noise injection adds noise perturbation in network weights [132]; data noise injection

directly inputs noise perturbation into data [24].

Metrics scaling uncertainty: Many metrics about uncertainty score are proposed based on the softmax vectors. As an important element in the sofmax vectors, winning score is proposed in [61]. Furthermore, temperature scaling [40] is proposed to get the calibrated probability by adding a scalar parameter to each class in calculating softmax vector. Applying winning score as prediction confidence is proposed in [40, 122]. This confidence is further applied in Expected Calibration Error [118], which is the absolute value of the difference between the accuracy and confidence of results. Besides, Overconfidence Error is proposed by applying winning score as confidence and penalizing samples with confidence values greater than accuracy values [151]. In addition, four metrics for result confidence are proposed in [161] by combining expectation and variance of predictions from different sampled parameter sets. In addition, cross-entropy is applied to calculate uncertainty score by dropout sampling and bin counting in [189], which also considers text classification with human involvement. Different from previous works, we improve the accuracy of uncertainty score by reducing the effect of overconfidence and considering three categories of uncertainty simultaneously.

Comparison with previous NLP works related to uncertainty. We compare MSD with some recent NLP works in terms of uncertainty categories. The comparison is listed in Table. 2.1. From the table, we can conclude that few works in NLP consider uncertainty in a comprehensive way (at least three categories of uncertainty are considered). Two works are noteworthy. Firstly, though we summarize [163] considers the structural uncertainty by two models with different layer designs, compared with NAS, which tries thousands of different neural architectures, which is not a good way to solve the structural uncertainty. Secondly, although [24] considers the same three categories of uncertainty in semantic parsing, the main differences between it and MSD are: (a) [24] only scales the uncertainty score in the tesing process, without interfering with the training process. While we apply both the training process and testing process for uncertainty score, because we assume the training process is more flexible and has more abundant information compared with the testing process. For examples, we apply the training process to keep the negative relationship between the training samples and their uncertainty by the mix-up. (b) Though we consider the three same categories of uncertainty, we have obvious difference in solving them. For examples, we solve the aleatoric uncertainty by the mix-up, while [24] applys Gaussian noise; we consider both the dropout and self-ensembling to solve pamametric uncertainty, while only dropout is considered in [24]. (c) The uncertainty scores between two works are calculated differently. We apply the reciprocal of winning scores as our uncertainty scores, while theirs is calculated by inputting confidence metrics to gradient tree boosting model [13].

Different overconfidence. Though [151] also mentions overconfidence, their overconfidence is different from ours. Their overconfidence is that the model accuracy is prone to be lower than what is indicated by the predictive score. While our overconfidence means that we cannot guarantee negative correlation between the winning scores and sample uncertainty, because the winning scores of training samples are all set as 1 by one-hot labels.

Table 2.1: Comparison between MSD and recent NLP works in terms of applied categories of uncertainty, where "A", "E", "P", and "S" represent aleatoric uncertainty, epistemic uncertainty, parametric uncertainty, and structural uncertainty respectively; "1" represents that the model considers respective uncertainty, while "0" represents that the model ignores the respective one.

Model	**Task**	**A**	**E**	**P**	**S**
[123]	Keyword extraction	0	0	1	0
[117]	Multi-modal classification	0	0	1	0
[44]	Image-caption retrieval	0	0	1	0
[67]	Entities of interest	0	1	1	0
[140]	Document quality assessment	0	0	1	0
[163]	Machine Translation	0	1	1	1
[161]	Machine Translation	0	0	1	0
[24]	Semantic Parsing	1	1	1	0
[190]	Semantic Parsing	0	0	1	0
[25]	Text classification	1	0	1	0
[93]	Text classification	1	0	0	0
[127]	Text classification	0	1	0	0
[154]	Text classification	0	1	1	0
[173]	Text classification	0	1	1	0
[189]	Text classification	0	0	1	0
MSD	Text classification	1	1	1	0

We name it also as overconfidence because the negative correlation is missing by the same winning scores, which should be designed differently. Though the winning scores of 1 means the highest confidence, we can make sample confidence different by mix-up, which will decrease the winning scores. Hence, compared with different but decreased winning scores, the original winning scores are overconfident. In other word, their overconfidence focuses on the two metrics of model performance, while ours focuses on the correlation between the winning scores and sample uncertainty, which also shows bias in the latent assumption.

2.3 Model

2.3.1 Basic Text Classification Model

In the traditional text classification model [141, 189], given an original text, we apply preprocessing (tokenization, lemmatization, etc.) to get its tokens in discrete numbers. Then, a pre-trained token embedding, such as word2vec [112] or Glove [131] is applied as a projector.

After that, a sequence of dense vectors for i-th text $\mathbf{Z}_i = [\mathbf{z}_{i1}, \mathbf{z}_{i2}, ..., \mathbf{z}_{in}]$ is derived by the embedding, where $\mathbf{z}_{ij}$ is the embedding of j-th word. The $\mathbf{Z}_i$ is fed to a sequence model f, such as CNN or RNN. Finally, we get i-th text representation $\mathbf{x}_i$ from the penultimate layer of f with dropout, and predicted semantic vector $\mathbf{y}_i = [y_{i1}, y_{i2}, ..., y_{ic}]$ from the last layer of f, where c is the number of classes and y_{ij} is the probability that i-th text belongs to j-th class. Finally, the f is trained by cross-entropy loss between the predicted semantic vector $\mathbf{y}_i$ and one-hot label $\widehat{\mathbf{y}}_i = [\widehat{y}_{i1}, \widehat{y}_{i2}, ..., \widehat{y}_{ic}]$ as follows,

$$L_{CE} = \sum_{j=1}^{c} \widehat{y}_{ij} log(y_{ij}). \tag{2.1}$$

In the testing process, uncertainty score U is formulated as follows,

$$U = \frac{1}{\max(\mathbf{y}_i^*)} \tag{2.2}$$

where $\mathbf{y}_i^*$ is semantic vector of i-th testing sample and $\max(\mathbf{y}_i^*)$ is the winning score of $\mathbf{y}_i^*$. Then, U conveys the uncertainty of model result.

2.3.2 Overview of MSD

Fig. 2.1 illustrates the training process of our model. In the first row, after prepossessing training text, we calculate the text representations, which is output of the penultimate layer with dropout. Then, we mix these representations in the batch-level. These mix-up-generated representations are fed into a fully connected (FC) layer for final semantic vectors. In the second row, we apply another model, which implements self-ensembling, with independent optimized parameters but the same structure as the one in the first row.

In the testing process, besides computing the reciprocals of winning scores with dropout mechanism, distinctiveness scores is also calculated by the Mahalanobis distance between the testing samples and distributions of training samples. Finally, the uncertainty score is calculated by adding the reciprocal of winning score and distinctiveness score.

2.3.3 MSD Training: Mix-up

Since the overconfidence is caused by the training samples with same winning scores due to the one-hot labels, and adding noise perturbation in the training process is a way to measure aleatoric uncertainty, we apply mix-up [151, 183] to jointly address the two issues. Mix-up generates new sample representations with various winning scores.

Concretely, we have i-th sample representation $\mathbf{x}_i$ from the penultimate layer of f with dropout. In a batch, we randomly mix i-th and j-th samples' representations ($\mathbf{x}_i$ and $\mathbf{x}_j$)

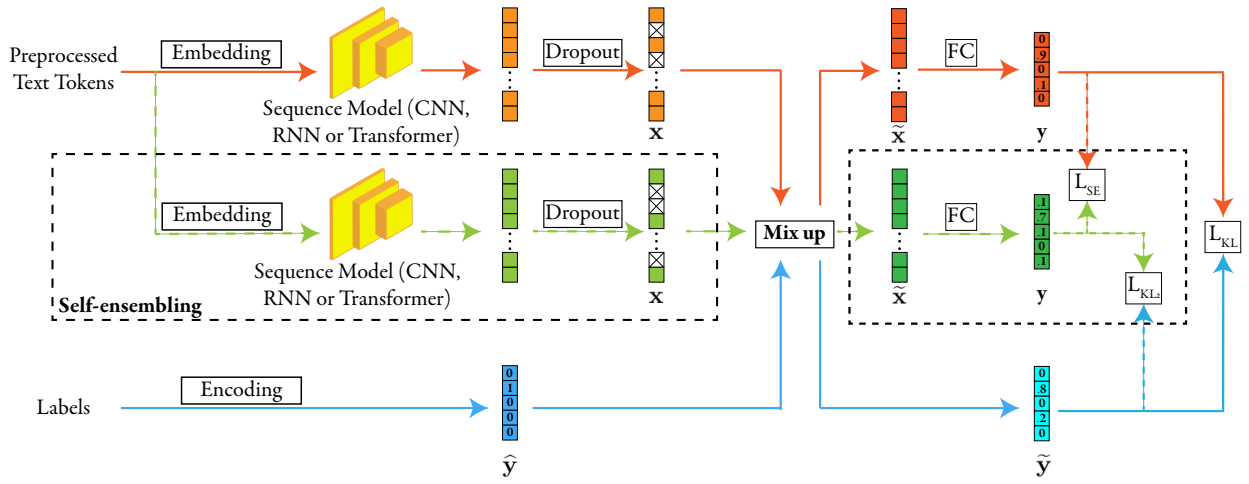

Figure 2.1: Diagram of training process of MSD. Orange arrows, green arrows, and blue arrows represent data flow of the first (default) model, second model, and labels respectively. Since self-ensembling is optional, it is illustrated as dotted lines. The distinctiveness score is not shown in the diagram since it is applied in the testing process. The numbers shown in $\mathbf{y}$, $\widehat{\mathbf{y}}$ and $\widetilde{\mathbf{y}}$ are probabilities of the semantic vectors.

and one-hot labels ($\widehat{\mathbf{y}}_i$ and $\widehat{\mathbf{y}}_j$) to get a mix-up sample representation $\widetilde{\mathbf{x}}$ and ground truth label $\widetilde{\mathbf{y}}$. We formulate mix-up as,

$$\widetilde{\mathbf{x}} = \alpha \mathbf{x}_i + (1 - \alpha)\mathbf{x}_j \tag{2.3}$$

$$\widetilde{\mathbf{y}} = \alpha \widehat{\mathbf{y}}_i + (1 - \alpha)\widehat{\mathbf{y}}_j \tag{2.4}$$

where α is a random number ranging from Ω to 1.00. The Ω is set above 0.5, so the i-th sample's semantics will be the main semantics of $\widetilde{\mathbf{x}}$, which is regarded as class of $\widetilde{\mathbf{x}}$ in MSD. Since the *difference* among winning scores and *negative correlation* between the winning scores and sample uncertainty are essential to reduce the impact of overconfidence, we analyze the two factors below.

Difference: Since $1 \geq \alpha \geq \Omega > 0.5$, $\widetilde{\mathbf{y}}$ has a winning score as α if $\mathbf{x}_i$ and $\mathbf{x}_j$ have different classes, or as 1 if two samples have the same class. Then, firstly, α or 1 is randomly chosen; secondly, the specific value of α is randomly sampled. Thus, different values of winning scores of training samples are achieved by the mix-up.

Negative correlation: Since $\widetilde{\mathbf{x}}$ includes i-th sample's representation $\mathbf{x}_i$ with ratio $\alpha > 0.5$, $\mathbf{x}_j$ can be regarded as noise of $\widetilde{\mathbf{x}}$. In one scenario, when $\mathbf{x}_i$ and $\mathbf{x}_j$ have different classes, $\mathbf{x}_j$ has obvious effect from noise on $\mathbf{x}_i$ due to different distributions in various semantics. In this case, when α is greater, $\widetilde{\mathbf{x}}$ has less noise from different semantic distributions. Then, $\widetilde{\mathbf{x}}$ is less adulterated and has higher confidence belonging to the class of $\mathbf{x}_i$. Since now winning score equals to α and the transitivity in math: the higher winning score is, the less adulterated $\widetilde{\mathbf{x}}$ is, which means $\widetilde{\mathbf{x}}$ has less uncertainty. In another scenario, where $\mathbf{x}_i$ and $\mathbf{x}_j$ belong to

same class, we assume that $\mathbf{x}_j$ has no effect from noise on $\mathbf{x}_i$, because $\mathbf{x}_j$ belongs to same distributions as $\mathbf{x}_i$ due to same semantics. Thus, the $\widetilde{\mathbf{x}}$ is the least uncertain in its class, and its winning score is the highest as 1. Hence, the negative correlation is boosted by mix-up.

After the mix-up, we feed $\widetilde{\mathbf{x}}$ rather than $\mathbf{x}$ to FC layer for its predicted semantic vector $\mathbf{y}$. However, we do not use cross-entropy loss (Eq. 2.1) in MSD, because it will learn the winning scores close to 1 due to no limit on the upper bound, which cannot ensure the negative correlation. Instead, we use KL divergence loss as one of our loss functions,

$$L_{KL} = \sum_{j=1}^{c} \widetilde{y}_{ij} log(\frac{\widetilde{y}_{ij}}{y_{ij}}) \tag{2.5}$$

because $\widetilde{\mathbf{y}}_i$ approximates to provide both upper and lower bound limitations by its non-zero element(s). The $\widetilde{y}_{ij}$ is a random value in each batch and each epoch. The overconfidence is reduced by mix-up due to *difference* and *negative correlation*. Besides, $\mathbf{x}_j$ can be regarded as random noise perturbation. The use of mix-up is an approximation of introducing noise into the data generation process. Thus, the level of mix-up is positively correlated with the aleatoric uncertainty of the generated data. Therefore, the aleatoric uncertainty is approximated by the mix-up, where the mix-up degree can be used as an approximation of the aleatoric uncertainty degree.

2.3.4 MSD Training: Self-ensembling

The parametric uncertainty comes from different sets of weights achieving similar training losses. Although the dropout can mitigate parametric uncertainty, previous works ignore the effect of self-ensembling [81, 129], which can boost the model combination and further decrease parametric uncertainty. We assume the dropout reduces the parametric uncertainty by loss generated within a model, while the self-ensembling reduces it from loss generated between models. The loss generated between models can help stabilize the model weights, because it can provide extra limitations besides the loss generated in a model, which reduce feasible weight sets. Plus, the designed component should aim at mitigating parametric uncertainty while have little impact on the model performance. Consider that the self-ensembling calculates the loss between the same models, which has more effect on model robustness and less impact on model performance, we apply the self-ensembling in addition to dropout to further mitigate the parametric uncertainty.

We construct another model with the same framework (e.g. layers, loss functions, dropout rate), and apply a self-ensemble loss L_{SE} to minimize the difference between two outputs from two models (the first model and the second model[4]) with the same framework and inputs,

$$L_{SE} = D[f_{\theta_1}(\widetilde{\mathbf{x}}, \phi_1), f_{\theta_2}(\widetilde{\mathbf{x}}, \phi_2)] \tag{2.6}$$

[4]The first model is our default model, and the second model is only required when we apply self-ensembling. They are shown in the first row and second row respectively in Fig. 2.1.

where θ_a is parameter set of a-th model, ϕ_a represents randomly sampled dropout neurons in neural network f, and $D[\mathbf{y_1}, \mathbf{y_2}]$ is a metric between two semantic vectors. D is Mean Square Error (MSE).

Although we already have the loss L_{SE}, we add KL divergence loss L_{KL_2} in the second model for the same setting. The L_{KL_2} is same as Eq. 2.5, while y_{ij} comes from the second model. We formulate MSD loss function L_{MSD} as follows,

$$L_{MSD} = L_{KL} + \lambda_1 L_{KL_2} + \lambda_2 L_{SE} \tag{2.7}$$

where λ_1 and λ_2 equal to 1 and a positive value respectively, when we apply self-ensembling, otherwise they both equal to 0. Together, the parametric uncertainty is further reduced.

2.3.5 MSD Testing: Distinctiveness Score

We also consider the epistemic uncertainty. Though out-of-distribution testing samples are known as the sources of epistemic uncertainty, they show that the epistemic uncertainty is the distinctiveness between the testing and training texts. However, it is not easy to consider the distinctiveness in the training process, because the training process is not aware of distributions of the testing samples. Therefore, we assume each class-level distribution of the training data can be modeled as a multivariable Gaussian distribution. We consider distance between a testing sample and each class-level Gaussian distribution as one part of the distinctiveness score. Motivated by [84], we apply Mahalanobis distance as follows,

$$m_{is} = (\mathbf{x}_i^* - \mu_s)^T \mathbf{\Sigma}^{-1} (\mathbf{x}_i^* - \mu_s) \tag{2.8}$$

where $\mathbf{x}_i^*$ is the representation of i-th testing sample in the first model without mix-up, and μ_s is the mean of representations of all training samples that belong to s-th class. $\mathbf{\Sigma}^{-1}$ is inverse of the covariance of all training samples. We do not apply the covariance in class-level to avoid singular matrices. After we obtain the Mahalanobis distance $\mathbf{m}_i = [m_{i1}, m_{i2}, ..., m_{ic}]$ of i-th testing sample to each class-conditional Gaussian distribution, we can also have a predicted class from this view, which is the class with the smallest distance in $\mathbf{m}_i$. In this way, we design penalty p as the other part in the distinctiveness score, which is not considered in [84], as below,

$$p_i = \begin{cases} 0 & r_m = r_y \\ \xi & r_m \neq r_y \end{cases} \tag{2.9}$$

where r_m is a classified result by $\mathbf{m}_i$ and r_y is the class with maximum probabilities in predicted semantic vector $\mathbf{y}_i^*$. ξ is a constant and set as 10 in our work. Our distinctiveness score d_i is,

$$d_i = log(\beta_1 \times p_i + \beta_2 \times min(\mathbf{m}_i)) \tag{2.10}$$

where log is a logarithm to the base 10; β_1 and β_2 are constants, both set as 1. Thus, the epistemic uncertainty is scaled in the uncertainty score to improve its accuracy. And the component improves CWS indirectly, because it remedies CWS for missing the epistemic uncertainty in the training process.

2.3.6 MSD Testing: Uncertainty Score

After we have trained our model by applying mix-up and self-ensembling, the winning scores will have higher confidence and accuracy due to reducing the overconfidence, aleatoric uncertainty, and parametric uncertainty. Regardless of whether we use self-ensembling or not, we only apply the first model to calculate the mean of predicted semantic vectors $\bar{\mathbf{y}}_i^*$ with dropout mechanism. Concretely, given a testing sample $\mathbf{x}_i^*$, we obtain k different predicted semantic vectors $\mathbf{y}_{i1}^*, \mathbf{y}_{i2}^*, ..., \mathbf{y}_{ik}^*$ by k times tryouts with the same dropout rate, from which, $\bar{\mathbf{y}}_i^*$ is the mean of k different $\mathbf{y}_i^*$. The maximum probability in $\bar{\mathbf{y}}_i^*$ is our winning score. Besides training for more confident winning scores, we also scale distinctiveness score d_i to measure the impact of epistemic uncertainty. We calculate our final uncertainty score U as,

$$U = \gamma_1 \times \frac{1}{\max(\bar{\mathbf{y}}_i^*)} + \gamma_2 \times d_i \tag{2.11}$$

where γ_1 and γ_2 are constants.

2.4 Experiments

Focusing on the text classification with human involvement, we evaluate the performance of MSD on four real-world datasets. Sec. 2.4.1 shows an overview of our experiment settings. Sec. 2.4.2 compares the performance between MSD and the state-of-the-art methods, and analyzes results of ablation experiments and parameter sensitivity analysis.

2.4.1 Experimental Setup

We apply Glove embedding [131], which is pretrained with dimension of 200, as our word embedding by default. For CNN model, we train MSD by setting a sequence model as a 3-layer CNN by default, with batch size of 32, momentum of 0.9, initial learning rate as 0.001 by Adam [75], kernel size of each layer as 3, 4, 5, respectively, as well as dropout rate of 0.3. For RNN model, Bidirectional Gated Recurrent Units (BiGRU) [66] is applied as an example of RNN model with two hidden layers. For Transformer, we apply XLnet [176] as an example[5].

Datasets

The four real-world-based datasets used in our experiments are as follow: (1) **20 Newsgroups** (20News) [83] includes 20 different news categories with 20,000 documents in it.

[5]More details and the experimental results on RNN and Transformer are shown in the appendix.

(2). **Amazon Reviews** (Amazon) [110] is a collection of reviews from Amazon from May 1996 to July 2013. For better comparison, we apply data from Sports and outdoors category, which is same as [189]. This dataset has 272,630 text samples with sentimental rating labels from 1 to 5. (3) **IMDb Reviews** (IMDb) has binary sentimental rating with 50,000 popular movie reviews. (4). **Yelp Reviews** (Yelp) [192] is a collection with sentimental rating labels from 1 to 5. It has two parts: the first part has 130,000 samples for each rating; the second part has 10,000 samples for each rating.

For the first three datasets, we apply the same split setting as [189], where for each dataset, 70% of samples form the training set, 10% of samples form the validation set, and the rest 20% form the testing set. For the Yelp dataset, we choose 9,000 samples randomly from the second part for each label as training set and the rest 1,000 samples for each label as the validation set, while all samples in the first part form the testing set.

Metrics

To evaluate the performance improvement of text classification with human involvement, which shows accuracy of uncertainty scores, we scale classification accuracy in different eliminated ratios. Concretely, for a testing set S with q samples and eliminated ratio r, we remove the most uncertain samples S_r from S based on uncertainty score ranking, where S_r has $r \times q$ samples. The more accurate uncertainty score we obtain, the more misclassified samples will be removed with the same r. Thus, if a model generates more accurate uncertainty scores, then the F1 scores for the rest testing samples will be higher with the same r. Because uncertainty score is more crucial for semantics with less training samples (e.g. “patient data samples" versus “the data for the healthy" in disease detection), we apply macro F1 score for the rest testing samples in the different eliminated ratios.

Baselines and Ablation Setting

We compare MSD with a state-of-the-art method, which achieves superior improvement of F1 scores in text classification with human involvement [189]. It proposes two methods: Dropout-Entropy (**DE**) is a dropout-entropy based model, and **DE**+**Metric** is a **DE** model along with metric learning. As for MSD, we divide MSD into three sub-models for ablation study: **MSD1** is a sub-model with only mix-up component; **MSD2-a** (abbreviate as **MSD2**) is one with two components, we apply mix-up and self-ensembling components by default; to show the flexibility of MSD, we design **MSD2-b**, which has two components as mix-up and distinctiveness score; and **MSD3** is one with all three components.

2.4.2 Experimental Results

Results of CNN model

Table 2.2, 2.3, 2.4, 2.5 report the F1 score improvement in the text classification with various eliminated ratios (10%, 20%, 30%, 40%) for CNN model. The improved ratios of F1 scores compared with no uncertainty elimination (0% column), are illustrated after the F1 scores. The parameter setting of Ω, λ_2, γ_1, γ_2 are given after each MSD in order. Three datasets (20 Newsgroups, Amazon, Yelp) are compared in macro F1 scores, except IMDb, which applies weighted F1 score for better comparison with [189]. From the tables, we conclude as below.

Table 2.2: Accuracy of uncertainty scores shown by improvement of macro F1 scores for the 20News (CNN model)

Methods (Ω, λ_2, γ_1, γ_2)	Uncertainty Ratio (Macro F1, Improved Ratio)				
	0%	**10%**	**20%**	**30%**	**40%**
DE	0.752	0.796(5.96%)	0.835(11.05%)	0.872(16.04%)	0.900(19.70%)
DE+Metric	0.774	**0.826**(6.70%)	**0.866**(11.97%)	**0.904**(16.87%)	**0.929**(20.02%)
MSD1 (1, 0, 1, 0)	0.751	0.808(**7.44%**)	0.854(13.50%)	0.894(**18.83%**)	0.923(**22.70%**)
MSD2 (1, 0.1, 1, 0)	0.760	0.812(6.92%)	0.849(11.73%)	0.886(16.59%)	0.920(21.47%)
MSD3 (1, 0.1, 1, 0.01)	0.760	0.812(6.95%)	0.856(**12.62%**)	0.889(16.98%)	0.921(21.22%)

1) **Better values of F1 scores:** MSDs (MSD1, MSD2, MSD3) improve F1 scores in values when certain portions of the most uncertain samples are eliminated. Especially for the Amazon dataset, the DE and DE+Metric both have negative growth when more uncertain samples are removed with eliminated ratios increased. This shows the accuracy of uncertainty score scaled in the testing is low for Amazon by DE and DE+Metric, while MSDs achieve significant increase on F1 when the most uncertain samples are eliminated, such as 26.64% increase in 40% elimination. In the 20News, MSDs achieve slightly lower F1 scores compared with DE+Metric, although slightly higher in F1 scores compared with DE. This is caused by obvious difference between texts with various semantics, so the uncertainty influence weakens and MSD is not very effective in the 20News.

Table 2.3: Accuracy of uncertainty scores shown by improvement of weighted F1 scores for the IMDb (CNN model)

Methods (Ω, λ_2, γ_1, γ_2)	Uncertainty Ratio (Weighted F1, Improved Ratio)				
	0%	**10%**	**20%**	**30%**	**40%**
DE	0.880	0.913(3.75%)	0.939(6.70%)	0.957(8.75%)	0.970(10.22%)
DE+Metric	0.884	**0.918**(3.85%)	**0.944**(6.79%)	0.961(8.71%)	0.974(10.18%)
MSD1 (1, 0, 1, 0)	0.874	0.907(3.87%)	0.933(6.79%)	0.952(8.95%)	0.967(**10.75%**)
MSD2 (1, 1, 1, 0)	0.883	**0.918**(3.92%)	**0.944**(6.82%)	0.961(8.85%)	**0.976**(10.46%)
MSD3 (1, 1, 1, 0.1)	0.882	**0.918**(**4.04%**)	0.943(**6.88%**)	**0.962**(**9.08%**)	0.974(10.49%)

2) **Better improved ratios of F1 scores:** If the uncertainty scores are more accurate,

higher improvement in the ratios of F1 scores would also be achieved. In comparison to DE and DE+Metric, MSDs always achieve better improved ratios of F1. Thus, MSDs generate more accurate uncertainty score. Especially, though MSD2 has lower F1 score compared with DE+Metric in 0% elimination, it still gets higher F1 score in 40% elimination in IMDb. Plus, though the F1 scores of MSDs are not higher compared with DE+Metric in 20News, higher improved ratios of F1 scores are achieved by MSDs. Thus, MSD is also competitive in comparison with baselines in 20News.

Table 2.4: Accuracy of uncertainty scores shown by improvement of macro F1 scores for the Amazon (CNN model)

Methods (Ω, λ_2, γ_1, γ_2)	**Uncertainty Ratio (Macro F1, Improved Ratio)**				
	0%	**10%**	**20%**	**30%**	**40%**
DE	0.438	0.447(2.07%)	0.439(3.15%)	0.438(1.39%)	0.428(-2.18%)
DE+Metric	0.432	0.443(2.56%)	0.439(1.60%)	0.431(-0.31%)	0.418(-3.27%)
MSD1 (1, 0, 1, 0)	0.434	0.458(5.40%)	0.463(6.52%)	0.464(6.76%)	0.472(8.73%)
MSD2 (1, 0.1, 1, 0)	0.453	**0.480**(5.83%)	**0.502**(10.67%)	0.505(11.38%)	0.530(17.01%)
MSD3 (1, 0.1, 1, 0.1)	0.435	0.467(**7.44%**)	0.490(**12.82%**)	**0.520**(**19.57%**)	**0.550**(**26.64%**)

3) **Effectiveness for each component by ablation setting:** Our proposed three components can be applied independently and further improve accuracy of uncertainty scores by combining them in the most situations, which shows effect of comprehensive consideration of uncertainty. In the 20News and Yelp datasets, when one or two components are added, we find consistent increase on F1 scores from MSD1 to MSD2, and from MSD2 to MSD3. Though the MSD3 does not achieve consistently higher improvement in various ratio eliminations in the IMDb and Amazon datasets, the performance of MSD2 is consistently higher than MSD1. It shows the effectiveness of self-ensembling in reducing the influence of uncertainty. Besides, the MSD3 achieves higher improvement of F1 scores in some eliminated ratios compared with MSD2 in the IMDb and Amazon datasets. We explain this as: the out-of-distribution testing texts do not distribute evenly in various eliminated ratios.

Table 2.5: Accuracy of uncertainty scores shown by improvement of macro F1 scores for the Yelp (CNN model)

Methods (Ω, λ_2, γ_1, γ_2)	**Uncertainty Ratio (Macro F1, Improved Ratio)**				
	0%	**10%**	**20%**	**30%**	**40%**
DE	0.562	0.583(3.61%)	0.598(6.44%)	0.614(9.16%)	0.629(11.84%)
DE+Metric	0.568	0.590(3.71%)	0.605(6.44%)	0.619(8.94%)	0.634(11.50%)
MSD1 (1, 0, 1, 0)	0.567	0.591(4.23%)	0.610(7.47%)	0.626(10.36%)	0.642(13.22%)
MSD2 (1, 0.1, 1, 0)	0.571	0.596(**4.41%**)	0.616(7.89%)	0.635(11.25%)	0.654(14.55%)
MSD3 (1, 0.1, 1, 0.01)	0.571	**0.597**(**4.41%**)	**0.617**(**7.94%**)	**0.636**(**11.30%**)	**0.655**(**14.63%**)

Results of Transformer and RNN model

Table 2.6 and Table 2.7 report the F1 score improvement in text classification with various eliminated ratios (10%, 20%, 30%, 40%) for BiGRU and XLnet respectively. Then, we conclude as below.

Table 2.6: Accuracy of uncertainty scores shown by improvement of macro F1 scores for the Amazon (BiGRU)

Methods (Ω, λ_2, γ_1, γ_2)	Uncertainty Ratio (Macro F1, Improved Ratio)				
	0%	**10%**	**20%**	**30%**	**40%**
DE	0.477	0.486(1.75%)	0.478(0.18%)	0.478(0.23%)	0.478(0.23%)
DE+Metric	0.471	0.478(1.50%)	0.466(-0.94%)	0.467(-0.84%)	0.466(-0.96%)
MSD1 (1, 0, 1, 0)	0.456	0.462(1.28%)	0.462(1.4%)	0.469(2.92%)	0.481(5.64%)
MSD2 (1, 0.1, 1, 0)	0.457	0.460(0.58%)	0.460(0.59%)	0.470(2.81%)	0.484(5.80%)
MSD3 (1, 0.1, 1, 0.1)	0.456	**0.497(8.88%)**	**0.524(14.95%)**	**0.531(16.39%)**	**0.508(11.36%)**

Table 2.7: Accuracy of uncertainty scores shown by improvement of macro F1 scores for the Amazon (XLnet)

Methods (Ω, λ_2, γ_1, γ_2)	Uncertainty Ratio (Macro F1, Improved Ratio)				
	0%	**10%**	**20%**	**30%**	**40%**
DE	0.422	0.422(0.00%)	0.428(1.38%)	0.423(0.26%)	0.424(0.38%)
DE+Metric	0.438	0.444(1.29%)	0.447(1.96%)	0.448(2.35%)	0.447(2.04%)
MSD1 (1, 0, 1, 0)	0.426	0.442(3.85%)	0.446(4.80%)	0.452(6.14%)	0.439(3.22%)
MSD2-a (1, 0.01, 1, 0)	0.415	0.436(5.03%)	0.440(6.06%)	0.434(4.46%)	0.422(1.56%)
MSD2-b (1, 0, 1, 1)	0.424	**0.451**(6.22%)	**0.470**(10.87%)	0.486(14.89%)	0.501(17.99%)
MSD3 (1, 0.01, 1, 1)	0.417	0.447(**7.16%**)	0.467(**11.96%**)	**0.487(16.81%)**	**0.509(21.95%)**

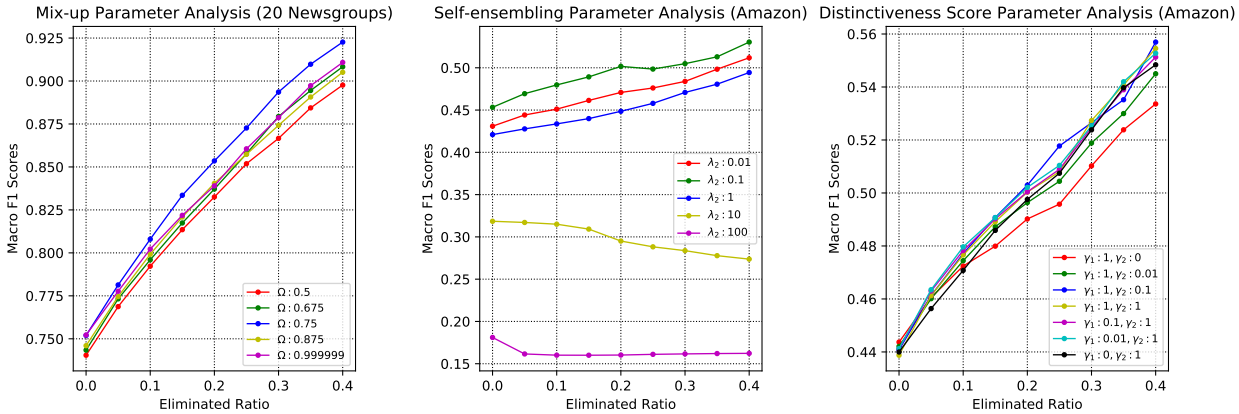

Figure 2.2: Diagrams for parameter sensitive analysis. The left panel shows how mix-up parameter Ω affects F1 scores. Middle panel shows how the self-ensembling parameter λ_2 affects F1 scores. Right panel shows how changes in γ_1 and γ_2 for distinctiveness score affect F1 scores.

1) **Higher performance in macro F1 by MSD3:** From Tables 2.6 and 2.7, MSD3 achieves higher improved ratios of F1 scores in different eliminated ratios. Though the MSD2-b has higher F1 scores with eliminated ratios 10% and 20%, the other F1 scores of MSD3 in the two tables are still the highest in each eliminated ratio. The superior improvement of both F1 scores and ratios of F1 scores shows the joint effect of three components. Furthermore, the results of MSD2-b and MSD3 show the effect of distinctiveness scores for macro F1 in Amazon, which has imbalanced data distributions. Besides, though MSD2-a performs poorly by mix-up and self-ensembling, this performance is reasonable. Because XLnet is a pretrained model, we have parameters of only two FC layers to train, which has much less feasible solutions of possible parameters compared with the CNN and RNN models. Thus, further decrease of feasible solutions of possible parameters brings negative effect in this case.

2) **Flexibility of MSD:** From the Table 2.6 and Table 2.7 for RNN and Transformer respectively, as well as Tables 2.2, 2.3, 2.4, and 2.5 for CNN model, we can observe the competitive performance of MSD in text classification F1 scores compared with two baselines. This verifies that MSD is effective to assemble with other DNNs (CNN, RNN and Transformer). Besides, the ablation setting of MSD1, MSD2-a, MSD2-b and MSD3 shows that the three components in MSD can be assembled arbitrarily based on the characteristics of datasets.

Parameter Sensitivity Analysis

The impact of different Ω for the mix-up, various λ_2 for the self-ensembling and γ_1, γ_2 for the distinctiveness score is discussed as below.

1) **Parameters for mix-up:** The left panel in Fig. 2.2 shows the effectiveness of different Ω. We apply $\Omega = 0.999999$ to approximate no mix-up. From the subfigure, we find: (1) the F1 scores are slightly sensitive to different Ω, while the improved ratios of F1 scores are not sensitive to the change of Ω. (2) For the 20News, when $\Omega = 0.75$, the macro F1 scores are the highest in different ratios, which are higher than the F1 scores of $\Omega = 0.999999$. This shows the effectiveness of mix-up in improving the accuracy of uncertainty score.

2) **Parameters for self-ensembling:** The impact of self-ensembling parameter λ_2 is shown in the middle panel in Fig. 2.2. This panel shows: (1) the F1 scores and their improved ratios in various eliminated ratios are significantly sensitive to λ_2, especially when λ_2 is greater than 1. (2) For Amazon dataset, macro F1 scores are the highest when $\lambda_2 = 0.1$ rather than $\lambda_2 = 0.01$. This again verifies the effectiveness of self-ensembling in improving the accuracy of uncertainty score.

3) **Parameters for distinctiveness score:** The right panel in Fig. 2.2 shows the impact of various γ_1 and γ_2 for distinctiveness score. We can see that: (1) the F1 scores are slightly sensitive to different γ_1 and γ_2. (2) The F1 scores of $\gamma_1 = 1, \gamma_2 = 0.1$ is around 2% higher compared with those of $\gamma_1 = 1, \gamma_2 = 0$, and nearly 1% higher compared with those of $\gamma_1 = 0, \gamma_2 = 1$. This presents the effect of distinctiveness score in generating more accurate

uncertainty score.

Results of transformer and RNN model

Concretely, we apply XLnet [176] as an example of the Transformer on Amazon dataset. The XLnet is pretrained from [169] where it has 12 layers, 12 heads and dimension of hidden state as 768. We apply the feature from the last hidden state of the XLnet, followed by two trainable layers, which is added by us. The two trainable layers are FC1 layer (768→ 768) and FC2 layer (768→ 5), because of 5 sentimental labels in Amazon. Only for this task, we apply its own word embedding rather Glove embedding with dimension of 200. The Macro F1 and Micro F1 of XLnet on Amazon are shown as Table 2.8 and Table 2.9 respectively. To show the flexibility of MSD, we try MSD2-b, which has two components as mix-up and distinctiveness score, and apply MSD2-a to represent default MSD2 setting, where it only has mix-up and self-ensembling.

Table 2.8: Accuracy of uncertainty scores shown by improvement of macro F1 scores for the Amazon (XLnet)

Methods(Ω, λ_2, γ_1, γ_2)	**Uncertainty Ratio(Macro F1, Improved Ratio)**				
	0%	**10%**	**20%**	**30%**	**40%**
DE	0.422	0.422(0.00%)	0.428(1.38%)	0.423(0.26%)	0.424(0.38%)
DE+Metric	0.438	0.444(1.29%)	0.447(1.96%)	0.448(2.35%)	0.447(2.04%)
MSD1 (1, 0, 1, 0)	0.426	0.442(3.85%)	0.446(4.80%)	0.452(6.14%)	0.439(3.22%)
MSD2-a (1, 0.01, 1, 0)	0.415	0.436(5.03%)	0.440(6.06%)	0.434(4.46%)	0.422(1.56%)
MSD2-b (1, 0, 1, 1)	0.424	**0.451**(6.22%)	**0.470**(10.87%)	0.486(14.89%)	0.501(17.99%)
MSD3 (1, 0.01, 1, 1)	0.417	0.447(**7.16%**)	0.467(**11.96%**)	**0.487**(**16.81%**)	**0.509**(**21.95%**)

Table 2.9: Accuracy of uncertainty scores shown by improvement of micro F1 scores for the Amazon (XLnet)

Methods(Ω, λ_2, γ_1, γ_2)	**Uncertainty Ratio(Micro F1, Improved Ratio)**				
	0%	**10%**	**20%**	**30%**	**40%**
DE	0.682	0.711(4.18%)	0.737(8.10%)	0.762(11.76%)	0.783(14.88%)
DE+Metric	0.686	0.717(4.47%)	0.744(8.49%)	0.771(12.46%)	0.796(16.12%)
MSD1 (1, 0, 1, 0)	0.683	0.722(5.63%)	0.754(10.43%)	0.789(15.46%)	0.821(20.19%)
MSD2-a (1, 0.01, 1, 0)	0.684	**0.723**(**5.72%**)	**0.758**(**10.92%**)	**0.792**(**15.88%**)	**0.824**(**20.57%**)
MSD2-b (1, 0, 1, 1)	0.683	0.717(5.12%)	0.740(8.39%)	0.755(10.59%)	0.783(14.76%)
MSD3 (1, 0.01, 1, 1)	0.684	0.722(5.51%)	0.749(9.37%)	0.767(12.12%)	0.791(15.56%)

Besides, Bidirectional Gated Recurrent Units (BiGRU) [66] is applied as an example of RNN model with two hidden layers. The Macro F1 and Micro F1 of BiGRU on Amazon are shown as Table 2.10 and Table 2.11 respectively.

From Tables 2.8, 2.9, 2.10, 2.11, we can conclude:

Table 2.10: Accuracy of uncertainty scores shown by improvement of macro F1 scores for the Amazon (BiGRU)

Methods(Ω, λ_2, γ_1, γ_2)	Uncertainty Ratio(Macro F1, Improved Ratio)				
	0%	**10%**	**20%**	**30%**	**40%**
DE	0.477	0.486(1.75%)	0.478(0.18%)	0.478(0.23%)	0.478(0.23%)
DE+Metric	0.471	0.478(1.50%)	0.466(-0.94%)	0.467(-0.84%)	0.466(-0.96%)
MSD1 (1, 0, 1, 0)	0.456	0.462(1.28%)	0.462(1.4%)	0.469(2.92%)	0.481(5.64%)
MSD2 (1, 0.1, 1, 0)	0.457	0.460(0.58%)	0.460(0.59%)	0.470(2.81%)	0.484(5.80%)
MSD3 (1, 0.1, 1, 0.1)	0.456	**0.497**(**8.88%**)	**0.524**(**14.95%**)	**0.531**(**16.39%**)	**0.508**(**11.36%**)

Table 2.11: Accuracy of uncertainty scores shown by improvement of micro F1 scores for the Amazon (BiGRU)

Methods(Ω, λ_2, γ_1, γ_2)	Uncertainty Ratio(Micro F1, Improved Ratio)				
	0%	**10%**	**20%**	**30%**	**40%**
DE	0.712	0.747(4.97%)	0.780(9.60%)	0.783(9.96%)	0.783(9.97%)
DE+Metric	0.709	0.745(5.06%)	0.755(6.48%)	0.754(6.33%)	0.753(6.20%)
MSD1 (1, 0, 1, 0)	0.706	0.745(**5.60%**)	0.781(**10.63%**)	0.817(**15.82%**)	0.849(**20.30%**)
MSD2 (1, 0.1, 1, 0)	0.708	**0.748**(5.58%)	**0.783**(10.54%)	**0.819**(15.58%)	**0.850**(20.02%)
MSD3 (1, 0.1, 1, 0.1)	0.709	0.746(5.32%)	0.778(9.83%)	0.808(13.97%)	0.848(19.66%)

Higher performance in macro F1 by MSD3. From Tables 2.8 and 2.10, MSD3 always achieves higher improved ratios of F1 scores in different eliminated ratios. Though the MSD2-b has higher F1 scores with eliminated ratios 10% and 20%, the other F1 scores of MSD3 in the two tables are still the highest in each eliminated ratio. The superior improvement of both F1 scores and ratios of F1 scores shows the joint effect of three components. Furthermore, from the results of MSD2-b and MSD3, we realize the effect of distinctiveness scores for macro F1 in Amazon, which has imbalanced data distributions. In addition, though MSD2-a performs poorly by mix-up and self-ensembling, this performance is reasonable. Because XLnet is a pretrained model, we have parameters of only two FC layers to train, which has much less feasible solutions of possible parameters compared with the CNN and RNN models. Thus, further decrease of feasible solutions of possible parameters brings negative effect in this case.

Higher performance in micro F1 by MSD2(-a). From Tables 2.9 and 2.11, we find MSD2 and MSD2-a always achieve the highest results with little increase compared with the MSD1. This still shows the effect of mix-up and self-ensembling. Though MSD3 performs not competitively compared with MSD1 and MSD2, it still outperforms DE+Metric obviously in both F1 scores and improved ratios of F1 scores in BiGRU. The poor performance of MSD3 compared with MSD1 and MSD3 also shows the offsets in calculating uncertainty scores by summing two parts in the testing process. Thus, how to reduce epistemic uncertainty in the training process will be our future work to avoid the offsets.

Flexibility of MSD. From the four tables and the tables describing CNN model, we can observe the competitive performance of MSD in text classification F1 scores compared with

two baselines. This verifies that MSD is effective to assemble with other DNNs (CNN, RNN and transformer). Besides, the ablation setting of MSD1, MSD2(-a), MSD2-b and MSD3 shows that the three components in MSD can be assembled arbitrarily based on the characteristics of datasets.

More experiment settings

Computing infrastructure. We do all experiments on two GPUs, which both are GTX 1080Ti. The RAM in our machine is 64 GB.

Running time. For the experiments on 20News, IMDb, Amazon with saved hidden states from pretrained XLnet, the training can be completed in 2 hours, and testing can be completed in 30 minutes. While calculating and saving hidden states from pretrained XLnet for Amazon dataset takes around 14 hours. For the experiments on Amazon (CNN) and Amazon (BiGRU), the training can be finished in 6 hours, while testing is finished in 1 hour and 3 hours respectively. All training epoches are set as 4 and testing will repeat $k = 100$ times tryouts with the same dropout rate.

Number of parameters. We have 7 parameters totally. Concretely, the mix-up has 1 parameter: Ω, which is a lower boundary of the mix-up random ratio α; the self-ensembling has 2 parameters λ_1 and λ_2 which are the coefficients of the losses L_{KL_2} and L_{SE} respectively; the distinctiveness score has 4 parameters, where β_1 and β_1 are the coefficients of the penalty and Mahalanobis distance respectively, and γ_1 and γ_2 are the coefficient of the reciprocal of winning scores and distinctiveness scores respectively.

Evaluation metrics link. Though we have explained evaluation metrics in the experiment section, we provide our metrics code in "emnlp_eval.py" in our submitted zip file (downloading form "SupMat__Software " of EMNLP website), which is revised from [189] metrics code "drop_entropy_eval.py" [6]. We revise their metrics, entropy calculated by bin count, into ours, reciprocal of winning score but still with dropout mechanism.

Hyperparameter configurations for best-performing models. We give the concrete parameter setting, which obtains best performance in certain DNN and component combinations, after MSDs in the first column of each Table, the remaining parameters are constants, which have been introduced in the model section.

Method of choosing hyperparameter values. For each setting, we will choose the hyperparameters with the highest micro F1 values (without elimination) in the validation process. This can help us find the hyperparameters related to the training process. Concretely, we firstly find a Ω from [0.675, 0.75, 0.875]. Then, we find a λ_2 from [0.001, 0.01, 0.1, 1, 10] based on the highest micro F1 values as well. As for the hyperparmeters related to the testing, we will run a test with only distinctiveness scores by setting $\gamma_1 = 0$ and $\gamma_2 = 1$,

[6] https://github.com/xuczhang/UncertainDC/blob/master/

to see current micro F1 and macro F1 values. If the mean of their values is greater than the one of MSD1, we will find γ_2 from 0.1, 1, or 10, else from 0.001, 0.01 or 0.1. Thus, a MSD3 on a task should try $3 + 5 + 1 + 3 = 12$ times.

Number of hyperparameter search trials. Based on the discussion in "Method of choosing hyperparameter values", and consider we have 6 tasks, we do 72 times hyperparmater search trials for MSDs.

2.5 Conclusion

We aims at generating more accurate uncertainty score to improve the performance of text classification with human involvement. We propose MSD with three independent components to improve the CWS by mitigating the effect of overconfidence and handling the impact of three categories of uncertainty. MSD can be applied to various DNNs (CNN, RNN, and Transformer) and each component in MSD can be arbitrarily assembled. Extensive experiments on four real-world datasets demonstrate that MSD obtains more accurate uncertainty scores, and superiorly improved classification performance when partial uncertain predictions are simulatively assigned to the experts.

Chapter 3

Uncertainty Estimation on Few-Shot Text Classification[1]

Few-shot text classification has extensive application where the sample collection is expensive or complicated. When the penalty for classification errors is high, such as early threat event detection with scarce data, we expect to know "whether we should trust the classification results or reexamine them." This paper investigates the Uncertainty Estimation for Few-shot Text Classification (UEFTC), an unexplored research area. Given limited samples, a UEFTC model predicts an uncertainty score for a classification result, which is the likelihood that the classification result is false. However, many traditional uncertainty estimation models in text classification are unsuitable for implementing a UEFTC model. These models require numerous training samples, whereas the few-shot setting in UEFTC only provides a few or just one support sample for each class in an episode. We propose Contrastive Learning from Uncertainty Relations (CLUR) to address UEFTC. CLUR can be trained with only one support sample for each class with the help of pseudo uncertainty scores. Unlike previous works that manually set the pseudo uncertainty scores, CLUR self-adaptively learns them using our proposed uncertainty relations. Specifically, we explore four model structures in CLUR to investigate the performance of three common-used contrastive learning components in UEFTC and find that two of the components are effective. Experiment results prove that CLUR outperforms six baselines on four datasets, including an improvement of 4.52% AUPR on an RCV1 dataset in a 5-way 1-shot setting.

[1]J.He, X. Zhang, S. Lei, F. Chen, A. Alhamadani, B. Xiao, C. Lu. CLUR: Uncertainty Estimation for Few-Shot Text Classification with Contrastive Learning. [C] In Proceedings of the 29th ACM SIGKDD international conference on knowledge discovery & data mining.

3.1 Introduction

Few-shot text classification learns a classifier using limited training texts [6, 115]. Few-shot scenarios often involve a crucial decision on whether or not to trust a model's results. For example, a state-of-the-art (SOTA) model diagnosing a new disease demands high accuracy but initially has access only to a few descriptions of the condition. One approach to achieve a higher classification accuracy is to recheck the most uncertain results by the experts [58,189]. Experts are expensive and scarce. Therefore, uncertainty estimation is pivotal in optimizing decision-making and saving expert resources in many few-shot applications. Here, we improve the accuracy of Uncertainty Estimation for Few-shot Text Classification (UEFTC). Specifically, UEFTC quantifies the likelihood of misclassification in scenarios with few samples [2]. UEFTC models should yield high uncertainty scores for misclassified predictions and low uncertainty scores for correct predictions.

However, the few-shot setting in UEFTC makes many uncertainty estimation methods difficult to use. Concretely, compared to traditional uncertainty estimation tasks, the few-shot setting in UEFTC provides only a few support samples or even one support sample (1-shot) per class in each episode [3]. Below, we describe how the current methods in uncertainty estimation cannot tackle UEFTC given the few-shot limitation and how our approach improves it.

The current uncertainty estimation methods are mainly of three kinds. First, Bayesian Neural Networks (BNN)-based methods learn a distribution over the model parameters [104, 125], or learn a distribution for each semantic class [9, 10, 145]. Due to the few-support-sample limitation in UEFTC, it is difficult to learn a distribution by a few or just one support sample per class. As a result, BNN-based methods are not suitable for addressing UEFTC. The second method is ensemble-based, which trains an uncertainty estimation model with augmentations of data (i.e., Gaussian noise [68], adversarial augmentation [130]) or structures (i.e., depth-based ensemble [5] and structure search [182]). The third method is pseudo-label-based, which uses the pseudo uncertainty scores as ground truth to learn an uncertainty model [58]. Since ensemble-based and pseudo-label-based methods do not require numerous support samples, we adopt these two methods to tackle UEFTC.

Though pseudo-label-based methods have growing application in recent years [11, 166], their current usage in uncertainty estimation [58] has a drawback of manually setting pseudo uncertainty scores as the ground truth. Concretely, [58] proposes MSD1 for uncertainty estimation of text classification, which manually sets coefficients of mix-up [183] as the pseudo uncertainty scores for an uncertainty model training (explained in Sec. 3.3.4). However, the manual-set pseudo uncertainty scores can be inaccurate because we do not know a training sample's ground-truth uncertainty score given a model structure. Due to the few-shot setting, the inaccurate pseudo uncertainty scores impact UEFTC more than tasks using numerous

[2] We detail the UEFTC task setting in Sec. 3.3.1

[3] The term "support" is explained in Sec. 3.3.1

samples because each sample weighs more in UEFTC. In a one-shot setting, an inaccurate pseudo uncertainty score means the unique support sample uses an inaccurate ground truth of uncertainty scores, leading to obvious training bias.

We propose Contrastive Learning from Uncertainty Relations (CLUR) to improve the accuracy of pseudo uncertainty scores. Instead of manually setting pseudo uncertainty scores, CLUR self-adaptively learns them by our proposed uncertainty relations. The uncertainty relations are either equal or unequal relations (i.e., $>$, $<$) between the uncertainty of a pair of augmented samples. The uncertainty relations are obtained from data augmentation. Since the usage of data augmentation is continuously optimized in contrastive learning, we design CLUR based on contrastive learning to better use the data augmentation. This is the first time that contrastive learning has been applied in UEFTC. Therefore, we also investigate whether the three commonly used model structures (detach, predictor, and intersection comparison that are introduced in Sec. 3.3.3) in contrastive learning are effective in UEFTC. Finally, we show that CLUR exceeds six baselines on four datasets. Our contributions are summarized below.

Improving uncertainty estimation by a few or just one support sample per class. To our knowledge, we are the first to solve UEFTC under its few-support-sample limitation. Our proposed CLUR can be trained with one support sample per class in each episode because it takes advantage of ensemble and pseudo-label-based methods. Our solution in UEFTC can also motivate uncertainty estimation in other few-shot applications.

Proposing and using uncertainty relations to self-adaptively learn pseudo uncertainty scores as the ground truth uncertainty. To address the issue of manually setting pseudo uncertainty scores, we generate augmented sample pairs to self-adaptively learn their pseudo uncertainty scores by our proposed uncertainty relations. Unlike current contrastive learning models that only have equal relations (i.e., having the same ($=$) or different ($\neq$) classes) between the augmented samples, our uncertainty relations include additional unequal relations, that are larger ($>$) or smaller ($<$) uncertainty relations among the augmented sample uncertainty.

Investigating the performance of the three common-used contrastive learning components in UEFTC. As the first study to apply contrastive learning in UEFTC, we also design four model structures in CLUR to investigate the performance of three common-used contrastive learning components in UEFTC. We find that two of them are effective in UEFTC, enabling us to optimize CLUR. Future UEFTC models can benefit from our findings.

Conducting extensive experiments and benchmarking the UEFTC. We demonstrate that CLUR effectively outperforms six baselines on four datasets (20News, RCV1, Amazon, and HuffPost), including an improvement of 4.52% AUPR on an RCV1 dataset in a 5-way 1-shot setting. We release our code as UEFTC benchmark.

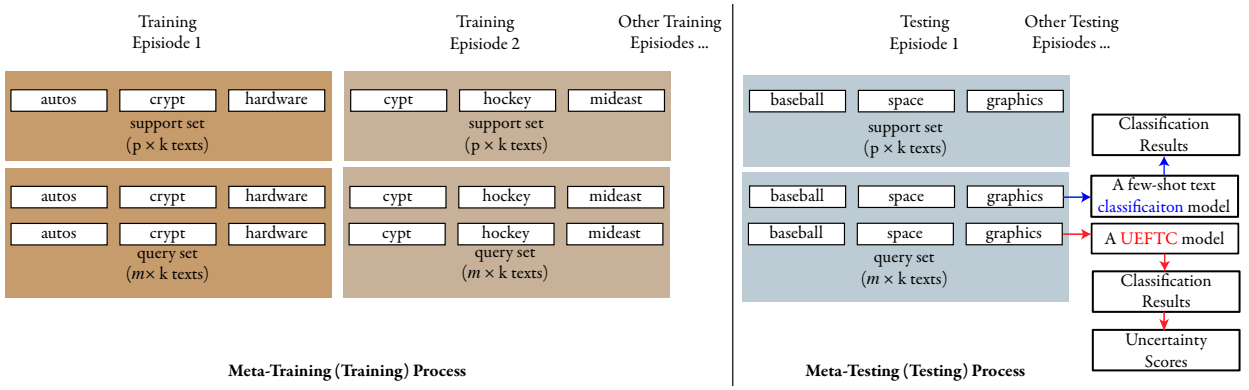

Figure 3.1: Diagram of UEFTC in sample splits, where $k = 3$, $p = 1$, and $m = 2$. Each tag in the diagram represents a text sample with a respective class. During the meta-training (training) process, a UEFTC model learns via the loss over the query samples in each training episode. The loss functions expect both accurate classification results and uncertainty scores. During the meta-testing (testing) process, the UEFTC model predicts a classification result and an uncertainty score for each query sample from each testing episode. We evaluate UEFTC by the performance of uncertainty scores of query sample classification results from each testing episode. The episodes drawn in this diagram are also applied to few-shot text classification. Compared to few-shot text classification, a UEFTC model additionally targets accurate uncertainty scores besides classification results.

3.2 Related Work

General methods for uncertainty estimation. There are mainly three uncertainty estimation methods: Bayesian Neural Network (BNN)-based [76], deep-ensemble-based [182], and pseudo-label-based [58]. BNN is a neural network with a prior distribution on model weights or dataset category distributions. As an approximation of BNN, Monte Carlo dropout [30, 73] uses dropout in the model in an ensemble way. Building on BNN, a recent study uses inducing matrices to assist in approximating posterior inference [137]. BNN can handle node classification as well [145]. Based on BNN, Evidential Neural Networks (ENN) [64, 87] calculate Dirichlet distributions, which also need numerous training samples to learn accurate distributions. The deep-ensemble-based method trains a fixed architecture with augmentations of data [68,130] or structures [5,182]. In addition, [31] scale seq2seq tasks by BNN and deep ensembles. As for the pseudo-label-based method [58], it generates the pseudo uncertainty scores for a training sample given a model. Since BNN-based methods usually require numerous samples, our CLUR is a combination of the deep-ensemble-based and the pseudo-label-based methods, where we use contrastive learning to connect these two methods.

Uncertainty estimation for text classification. It focuses either on the training or the testing data. For example, [165] annotate unlabeled samples with higher uncertainty for training. For testing, it mainly has two tasks: *OOD detection* [23,51,84] for predictions, such

as [63]; and *misclassified result detection*, where testing samples are in-domain, such as [189] and [58]. [189] use dropout sampling for uncertainty scores. Three modules are proposed in [58], where MSD3 calculates the sample distributions and is not applicable to UTFTC. Compared to them, UEFTC addresses misclassified result detection. Different from models requiring many training samples [58, 189], CLUR is trainable with one support sample per class in each episode. To our knowledge, we are the first to estimate uncertainty for few-shot text misclassification detection.

Few-shot text classification. Few-shot text classification has received increasing attention in recent years [88]. The few-shot text classification models are mainly in two categories: transfer-learning-based and meta-learning-based methods. The transfer-learning-based methods transfer well-learned knowledge to a new task, such as fine-tuning a pre-trained model by samples from a new task [42]. The meta-learning-based methods learn meta-knowledge of how to learn from a new task by meta-training episodes. Then the well-trained meta-knowledge is applied to a new task in meta-testing episodes for evaluation. UEFTC focuses on meta-learning-based methods. As a representative SOTA meta-learning-based method, FTC-DS [6] learns token embedding with the assistance of distributional signatures. Also mining assisted information, [33] dynamically update knowledge from base classes by a memory module. Meta-level attention is learned in LEA [62] based on pre-trained language models. In addition, MLADA [47] uses a generator and a discriminator to conduct adversarial learning for domain adaption in few-shot text classification. Since FTC-DS is a frequently used baseline and its attention-based token embedding is common among few-shot text classification, we use FTC-DS to study UEFTC.

Contrastive learning. Contrastive learning has been broadly applied in unsupervised representation learning [7, 14, 15, 38] by reducing the distance between positive pairs and enlarging the distance between negative pairs. Recently, contrastive learning has also been applied in supervised learning [74, 167] by positive and optional negative pairs, such as fine-tuning pre-trained language models [22, 39]. Many contrastive learning models use the detach, predictor, or intersection comparison components [7, 16, 38, 39, 92, 100, 144, 171]. How contrastive learning is used in supervised learning can be further divided into two categories: using negative samples and using no negative samples. For instance, [177] propose decoupled contrastive loss and [164] minimize the divergence between the weak and the strong augmented samples, they both need negative samples in the model training. As an example of using no negative samples, SimSiam [16] finds that using a Siamese net with detach operation achieves similar results by only a single sample in each update. Due to the few-support-sample limitation in UEFTC, contrastive learning using no negative samples [16] is more suitable, which further reduces the burden of the required sample size. Besides, unlike previous contrastive learning models [14, 16, 38, 164, 177], which only have equal relations between the augmented samples, our proposed uncertainty relations have additional unequal relations. Their equal relations are samples having the same ($=$) or different ($\neq$) classes, but our unequal relations are larger ($>$) or smaller ($<$) uncertainty relations between the sample uncertainty.

3.3 Preliminary Knowledge

3.3.1 UEFTC Task Settings

Problem Statement. As shown in Fig. 3.1, besides classifying texts like a few-shot text classification model, UEFTC additionally estimates the uncertainty scores for the classification results so that we can decide whether to trust the model prediction or not. A UEFTC model aims to learn how to acquire knowledge from training samples among training classes L^{Tr} during the meta-training. Then, given new classes L^{Te} during meta-testing, which are disjoint from L^{Tr}, a well-trained UEFTC model can quickly learn how to predict classes of testing samples among L^{Te} and their uncertainty scores. A better UEFTC model not only achieves higher classification performance but also gives higher uncertainty scores for misclassified results and lower uncertainty scores for correct results.

Meta-training. To meta-train a few-shot text classification model or a UEFTC model Θ, we create training episodes, shown as Fig. 3.1. Among a p-shot k-way setting, each training episode is built by randomly sampling k classes from L^{Tr}. From each of these k classes, we randomly sample p samples as a training set and m samples as a testing set. We update Θ based on loss over these testing samples in each training episode. We also repeat this model update among other p-shot k-way training episodes. In [6, 158], the training set of a training/testing episode is known as the support set, and its testing set is called the query set. Given the support set, we call Θ as a p-shot k-way few-shot model. According to different tasks, Θ can be a p-shot k-way few-shot text classification model or a p-shot k-way UEFTC model.

Meta-testing. To meta-test Θ, we use the same p-shot k-way setting to extract testing episodes from L^{Te}, shown as Fig. 3.1. For each testing episode, we use $p \times k$ support samples to update the model, which is further evaluated by the $m \times k$ query samples. A few-shot text classification model is evaluated by the classification accuracy on testing query sets. Besides the classification performance, we further evaluate a UEFTC model by the accuracy of estimated uncertainty scores of classification results on testing query sets. We aim to learn a UEFTC model predicting more accurate uncertainty scores for the classification results of testing query samples.

3.3.2 Few-Shot Text Classification Model

A few-shot text classification model aims at predicting unseen classes of query samples in testing episodes. Each episode [4] samples a support set and a query set randomly. We build CLUR on a SOTA model called FTC-DS [6], which uses attention-based tokens that are common-used in few-shot text classification [32, 34, 69, 147]. In each episode, FTC-DS

[4] An "episode" without defining a training or testing episode applies to both training and testing episodes. And a "sample" without defining a support or query sample is also applicable to both.

first gets token sequences in discrete numbers for both support and query samples. It then applies a frozen pre-trained token embedding to represent the support and query texts. Thus, a sequence of vectors $\mathbf{Z} = [\mathbf{z}_1, \mathbf{z}_2, ..., \mathbf{z}_n]$ for a text is obtained by the embedding, where $\mathbf{z}_i$ is the embedding of the i-th word. For a text embedding $\mathbf{x}$, FTC-DS sums each token embedding $\mathbf{z}$ as,

$$\mathbf{x} = \sum_i \alpha_i \cdot \mathbf{z}_i, \tag{3.1}$$

where $\alpha = [\alpha_0, \alpha_1, ..., \alpha_n]$ is the attention to learn. Then, FTC-DS gets a projection matrix $\mathbf{W}$, which is the solution of a regression loss $L = ||\mathbf{X}_S\mathbf{W} - \mathbf{Y}_S||_F^2 + \lambda||\mathbf{W}||_F^2$. To solve it, we have $\mathbf{W} = \mathbf{X}_S^T(\mathbf{X}_S\mathbf{X}_S^T + \lambda\mathbf{I})^{-1}\mathbf{Y}_S$, where $\mathbf{X}_S$ and $\mathbf{Y}_S$ are the text embeddings and labels of all support samples in an episode, $\mathbf{I}$ is an identity matrix, while λ is a scalar to be learned. Finally, FTC-DS predicts semantic vectors for a query set by a projector g as,

$$\widetilde{\mathbf{Y}}_Q = g(\mathbf{X}_Q) = a\mathbf{X}_Q\mathbf{W} + b \tag{3.2}$$

where $\{\cdot\}_Q$ has similar mean to $\{\cdot\}_S$, but $\{\cdot\}_Q$ refers to query samples. The model output $\widetilde{\mathbf{Y}}_Q \in \mathbb{R}^{m\times k}$ is a sequence of semantic vectors in an episode. The a and b in Eq. 3.2 are learnable scalars. In each training episode, FTC-DS updates α, λ, a, b by cross-entropy loss L_{CE} between model output $\widetilde{\mathbf{Y}}_Q$ and query set labels $\mathbf{Y}_Q$. In each testing episode, FTC-DS uses Eq. 3.2 to get $\widetilde{\mathbf{Y}}_Q$ for model evaluation.

3.3.3 SimSiam

SimSiam [16] is a SOTA contrastive learning model that only uses positive pairs for representation learning. The other SOTA contrastive learning models additionally require numerous negative pairs or large batch sizes, while SimSiam does not. Therefore, SimSiam is more suitable for UEFTC than other SOTA models as it requires fewer training samples. Plus, SimSiam also uses the three common-used contrastive learning components. As a result, our usage of data augmentation in CLUR is motivated by SimSiam. We briefly introduce SimSiam and the three components.

Given a sample $\mathbf{t}$, SimSiam augments it twice. The augmented samples $\mathbf{t}_1$ and $\mathbf{t}_2$ from $\mathbf{t}$ are input to a projector, which outputs projections $\widetilde{\mathbf{y}}_1 \in \mathbb{R}^k$ and $\widetilde{\mathbf{y}}_2 \in \mathbb{R}^k$ respectively, where k is class number. Finally, the projections $\widetilde{\mathbf{y}}_1$ and $\widetilde{\mathbf{y}}_2$ are input to a predictor, which outputs predictions $\widehat{\mathbf{y}}_1 \in \mathbb{R}^k$ and $\widehat{\mathbf{y}}_2 \in \mathbb{R}^k$ respectively. Its loss function is,

$$L_{SimSiam} = D[\widehat{\mathbf{y}}_1, o(\widetilde{\mathbf{y}}_2)] + D[\widehat{\mathbf{y}}_2, o(\widetilde{\mathbf{y}}_1)] \tag{3.3}$$

where D is cosine similarity and o is the *detach* (DT) operation to stop the gradient [16, 38, 144, 171]. D scales between $\widetilde{\mathbf{y}}$ and $\widehat{\mathbf{y}}$, which is an *intersection comparison* (IT) [16, 38, 92, 100]. Projections $\widetilde{\mathbf{y}}$ can be used for model inference, but many contrastive learning models extra design the *predictor* (PD) [7, 15, 38, 39] and use its predictions $\widehat{\mathbf{y}}$ for inference. Since DT, IT, and PD are common in contrastive learning, we investigate whether they improve CLUR. Our findings can benefit the design of future UEFTC models with data augmentation.

3.3.4 MSD: pseudo uncertainty

MSD1 [58] uses the mix-up to augment data and simulate the generation of uncertainty. It has an augmented sample $\widetilde{\mathbf{x}}$ by two text embedding $\mathbf{x}_i$ and $\mathbf{x}_j$ with their respective labels $\mathbf{y}_i$ and $\mathbf{y}_j$ as below,

$$\widetilde{\mathbf{x}} = \vartheta \mathbf{x}_i + (1-\vartheta)\mathbf{x}_j, \widetilde{\mathbf{y}} = \vartheta \mathbf{y}_i + (1-\vartheta)\mathbf{y}_j \tag{3.4}$$

where ϑ is a random number ranging from Ω to 1. The Ω is set above 0.5. It then learns a KL divergence loss for the augmented data. It uses a basic way to measure uncertainty, which is the reciprocal of the maximum probability of a softmax vector. Thus, its pseudo uncertainty is $\frac{1}{\vartheta}$ because of $\widetilde{\mathbf{y}}$, which is used as the pseudo uncertainty score to train an uncertainty estimation model. However, the manual-set pseudo uncertainty scores are inaccurate, because we have implicit knowledge of the pseudo uncertainty scores.

3.4 Our Model: CLUR

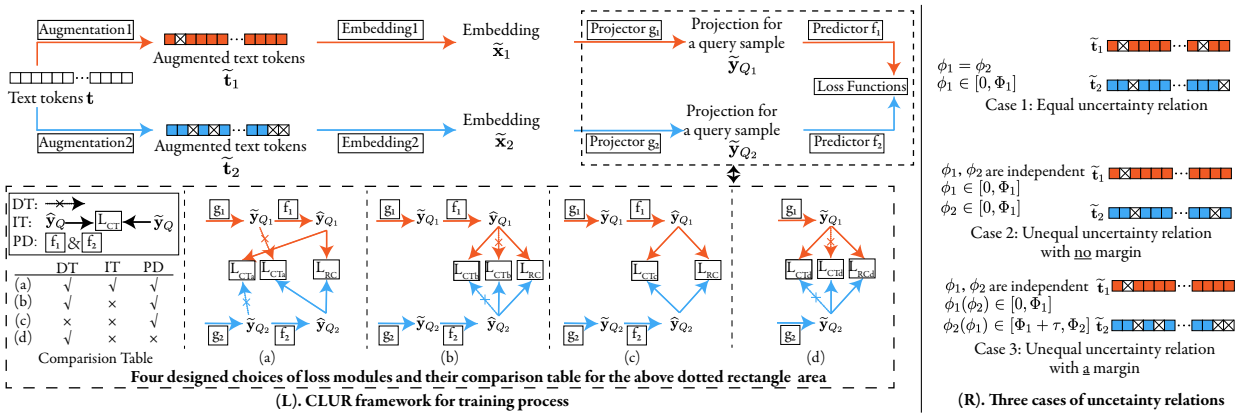

Figure 3.2: The left diagram (Fig. 3.2(L)) shows the training process of CLUR. The right diagram (Fig. 3.2(R)) shows three cases of uncertainty relations. Red and blue represent the data related to the first and second submodels respectively. The bottom dotted rectangle in Fig. 3.2(L) details four choices of loss modules and their comparison table, where "✓" ("×") means the design is used (unused) in the respective loss module. Fig. 3.2(R) illustrates our three cases of uncertainty relations by an example, where the example has $\Phi_1 = \tau = \Phi_2 = \frac{2}{n}$. Case 1 claims $\widetilde{\mathbf{t}}_1$ and $\widetilde{\mathbf{t}}_2$ have the same uncertainty; case 2 and case 3 both claim that $\widetilde{\mathbf{t}}_1$ has smaller uncertainty than $\widetilde{\mathbf{t}}_2$, where we verify the case 3 has more accurate pseudo uncertainty relations due to τ by our results in Sec. 3.5.2 and analysis in Sec. 3.4.6. During the testing process, we only use the first submodel and skip the "augmentation1" module to get the classification results and their uncertainty scores.

3.4.1 Overview Of CLUR

The upper panel of Fig. 3.2(L) shows the training process of our UEFTC model, Contrastive Learning from Uncertainty Relations (CLUR). CLUR is a pseudo-Siamese net [172], having two identical submodels with the same structure but different weights. In the first row (first submodel) of the upper panel, we augment the texts from a support set and a query set in each training episode. We then get the text embeddings for the support and query sets by the "embedding1" module. The query text embeddings are input to a projector g_1 to get their projections. These query text projections are further input to a predictor f_1 for their predictions. Similarly, the other projections and predictions for the same query set are obtained by the second submodel, shown in the second row with blue arrows in Fig. 3.2(L). Moreover, we design four choices of loss modules to train CLUR, shown in the bottom panel of Fig. 3.2(L). The four choices of loss modules verify whether DT, IT, and PD (defined in Sec. 3.3.3) help improve UEFTC. During the testing process, CLUR only uses the first submodel and skips the "augmentation1" module to get query text classification and uncertainty estimation results.

3.4.2 CLUR Training: Uncertainty Relations

As we don't know the true pseudo uncertainty scores, manually setting them can be inaccurate. Instead, CLUR self-adaptively learns the pseudo uncertainty scores via our uncertainty relations. Below, we introduce the augmentation module, which generates the uncertainty relations.

Augmentation module. Our data augmentation method is token-mask [175], which randomly masks the tokens in a text. We choose this data augmentation method, as it only needs one original sample, satisfying UEFTC's few-support-sample limitation.

Uncertainty Relations. Given an n-word text from either a support set or a query set, we have its token vector $\mathbf{t} \in \mathbb{R}^n$ in discrete numbers. We randomly mask $\mathbf{t}$ twice and obtain two different augmented token vectors $\widetilde{\mathbf{t}}_1 \in \mathbb{R}^n$ and $\widetilde{\mathbf{t}}_2 \in \mathbb{R}^n$ as below,

$$\begin{aligned} \widetilde{\mathbf{t}}_1 &= \mathbf{t} \cdot \mathbf{m}_{\phi_1} \\ \widetilde{\mathbf{t}}_2 &= \mathbf{t} \cdot \mathbf{m}_{\phi_2} \end{aligned} \tag{3.5}$$

where $\mathbf{m}_{\phi_1}$ and $\mathbf{m}_{\phi_2}$ are binary vectors to randomly mask $\mathbf{t}$ by ratios of ϕ_1 and ϕ_2, respectively. The ratios ϕ_1 and ϕ_2 are random numbers between 0 and 1 for each sample in each epoch. The different cases of ϕ_1 and ϕ_2 generate various uncertainty relations. We list three cases below, where their examples are shown in Fig. 3.2(R).

Case 1: Equal uncertainty relation. $\phi_1 \in [0, \Phi_1]$ and $\phi_1 = \phi_2$, where Φ_1 is a boundary of ϕ_1. In case 1, binary-mask vectors $\mathbf{m}_{\phi_1}$ and $\mathbf{m}_{\phi_2}$ have the same numbers of 0s and 1s but in random order. Since $\phi_1 = \phi_2$ in case 1, a pair of augmented samples from a text has the same numbers of the masked tokens. Due to their same numbers of masked tokens, case 1 assumes $\widetilde{\mathbf{t}}_1$ and $\widetilde{\mathbf{t}}_2$ have the same uncertainty. Thus, data augmentation in case 1 generates

equal uncertainty relations.

Case 2: Unequal uncertainty relation without a margin. In case 2, $\phi_1 \in [0, \Phi_1]$, $\phi_2 \in [0, \Phi_1]$, ϕ_1 and ϕ_2 are independent. Then, the number and order of 0s and 1s in $\mathbf{m}_{\phi_1}$ and $\mathbf{m}_{\phi_2}$ can both be different. Since a text with more masked tokens is harder for a model to predict, the model prediction is more uncertain. For a pair of augmented texts, case 2 regards an augmented text with more masked tokens as more uncertain than the other augmented text with fewer masked tokens. Compared to case 1, case 2 has more data diversity to provide more information. This is because case 1 limits $\phi_1 = \phi_2$, but ϕ_1 and ϕ_2 in case 2 are independent, with $\frac{1}{n}$ probability that $\phi_1 = \phi_2$ and $\frac{n-1}{n}$ probability that $\phi_1 \neq \phi_2$.

Case 3: Unequal uncertainty relation with a margin. Case 3 is designed to solve the issues in cases 1 and 2. Specifically, case 1 assumes a pair of texts with the same number of masked tokens has equal uncertainty. But each token has a different contribution to the text classification. Second, even though we have $\phi_1 < \phi_2$ or $\phi_1 > \phi_2$ in case 2, the divergence between the numbers of masked tokens might be too small to crucially impact the uncertainty relations. Though each token contributes differently to text semantics, a larger difference in the masked token numbers leads to a more accurate pseudo uncertainty relation [5]. Thus, case 3 sets a margin τ to enlarge the divergence in the masked token numbers among a pair of augmented samples. Specifically, case 3 has a 50% chance that $\phi_1(\phi_2) \in [0, \Phi_1]$ and $\phi_2(\phi_1) \in [\Phi_1 + \tau, \Phi_2]$, where Φ_2 is a boundary and the other 50% chance is shown in the brackets. In case 3, one augmented sample has at most $\Phi_1 \times n$ masked tokens, and the other one has at least $(\Phi_1 + \tau) \times n$ masked tokens.

In the three cases, we set $\Phi_1 \in (0, 0.5]$ and $\Phi_2 \in (0, 0.5]$ to avoid losing much text context. Any above uncertainty relations are then used to learn pseudo uncertainty scores described in Sec. 3.4.4.

Advantage of uncertainty relations. In short, it is easier to manually set pseudo uncertainty relations than to manually set pseudo uncertainty scores. Concretely, the uncertainty relations have only three possible values ($=$, $>$, $<$), but the uncertainty scores have countless possible values (any number$>$0). Further, when the divergence of a pair of augmented samples is enlarged (such as enlarged by τ in case 3), the uncertainty relations are more explicit to us, but the knowledge about pseudo uncertainty scores does not increase. Thus, the much fewer possible values and more observable change in setting uncertainty relations make it easier than setting pseudo uncertainty scores.

3.4.3 CLUR Training: General Modules

Embedding module. As shown in the upper panel of Fig. 3.2(L), after getting a pair of augmented token vectors $\widetilde{\mathbf{t}}_1$ and $\widetilde{\mathbf{t}}_2$ by a chosen case for either a support or query text in training episodes, we use the frozen pre-trained word embedding to get their token embeddings $\widetilde{\mathbf{Z}}_1 \in \mathbb{R}^{n \times u}$ and $\widetilde{\mathbf{Z}}_2 \in \mathbb{R}^{n \times u}$, where u is the dimension of word embedding. Sim-

[5] Detailed analysis is in Sec. 3.4.6

ilar to Eq. 3.1, we then accumulate each token embedding in $\widetilde{\mathbf{Z}}_1$ by learnable attentions $\widetilde{\alpha}_{\mathbf{1}} = [\widetilde{\alpha}_{11}, \widetilde{\alpha}_{12}, ..., \widetilde{\alpha}_{1n}]$ to get the text embedding $\widetilde{\mathbf{x}}_1 \in \mathbb{R}^u$, where $\widetilde{\mathbf{x}}_1 = \sum_{i=1}^{n} \widetilde{\alpha}_{1i} \cdot \widetilde{\mathbf{z}}_{1i}$. Similarly, we get another text embedding $\widetilde{\mathbf{x}}_2 \in \mathbb{R}^u$ from $\widetilde{\mathbf{Z}}_2$.

Projector module. We then use a projector g_1 like Eq. 3.2 to get our projection $\widetilde{\mathbf{y}}_{Q_1} \in \mathbb{R}^k$ for a query sample, by

$$\widetilde{\mathbf{y}}_{Q_1} = g_1(\widetilde{\mathbf{x}}_{1Q}) = a_1 \widetilde{\mathbf{x}}_{1Q} \mathbf{W}_1 + b_1 \tag{3.6}$$

where $\widetilde{\mathbf{x}}_{1Q}$ is an augmented query text embedding vector from the first submodel, a_1 and b_1 are the learnable parameters. For $\mathbf{W}_1$, it is calculated as,

$$\mathbf{W}_1 = \widetilde{\mathbf{X}}_{1S}^T (\widetilde{\mathbf{X}}_{1S} \widetilde{\mathbf{X}}_{1S}^T + \lambda_1 \mathbf{I})^{-1} \mathbf{Y}_S \tag{3.7}$$

where $\widetilde{\mathbf{X}}_{1S}$ is the augmented support texts' embedding tensor from the first submodel. λ_1 is a learnable parameter. The $\mathbf{W}_1$ in Eq. 3.7 is the solution of a regression loss, $L_1 = ||\widetilde{\mathbf{X}}_{1S} \mathbf{W}_1 - \mathbf{Y}_S||_F^2 + \lambda ||\mathbf{W}_1||_F^2$. Similar to Eq. 3.6 and Eq. 3.7, we get $\widetilde{\mathbf{y}}_{Q_2} \in \mathbb{R}^k$ by g_2, which has the same structure as g_1 but with different parameters.

3.4.4 CLUR Training: Explored Structures

Predictor module. The uncertainty relations are obtained from data augmentation. CLUR adopts contrastive learning in which the usage of data augmentation has been continuously optimized. Like the contrastive learning methods commonly do, the projection $\widetilde{\mathbf{y}}_{Q_1}$ is then input to a predictor f_1. The output of f_1 is the prediction of CLUR, $\widehat{\mathbf{y}}_{Q_1} \in \mathbb{R}^k$. Similarly, we get $\widehat{\mathbf{y}}_{Q_2} \in \mathbb{R}^k$ by f_2.

Loss Modules. To investigate the effects of three common-used contrastive learning modules (detach, intersection comparison, and predictor introduced in Sec. 3.3.3) in CLUR, we design four loss modules based on the projections $\widetilde{\mathbf{y}}_{Q_1}$, $\widetilde{\mathbf{y}}_{Q_2}$ and predictions $\widehat{\mathbf{y}}_{Q_1}$, $\widehat{\mathbf{y}}_{Q_2}$. The bottom panel in Fig. 3.2(L) shows the designed loss module choices. Only contrastive loss L_{CT_a} (case 1 in Sec. 3.4.2, Eq. 3.9) in loss choice (a) is similar to SimSiam. All the other loss choices are our original designs. Loss (b) has the best performance among the four loss module choices in UEFTC by removing the component "intersection comparison". (c) shows the performance of "detach". And (d) verifies the effectiveness of the "predictor".

Loss module (a). We first design a revised cross-entropy loss L_{RC} to keep classification performance and calibrate the prediction $\hat{\mathbf{y}}_Q$ for uncertainty estimation. In a training episode, a query text with one-hot label $\mathbf{y}_Q \in \mathbb{R}^k$ has its L_{RC} as,

$$\begin{aligned} L_{RC} = &\max\{L_{CE}(\widehat{\mathbf{y}}_{Q_1}, \mathbf{y}_Q) + \log(\beta), 0\} + \\ &\max\{L_{CE}(\widehat{\mathbf{y}}_{Q_2}, \mathbf{y}_Q) + \log(\beta), 0\} \end{aligned} \tag{3.8}$$

where L_{CE} is the traditional cross-entropy loss between the prediction $\widehat{\mathbf{y}}_Q$ and one-hot label $\mathbf{y}_Q$. We add $\log(\beta)$, $\beta \in [0.5, 1)$ to each L_{CE}, which has a penalty of 0 for L_{RC} once the probability for the correct class is above β. With our L_{RC}, the probability for the correct

class in $\widehat{\mathbf{y}}_Q$ is not always close to 1, but has feasible solutions in a larger range $[\beta, 1)$, this is further explained in Sec. 3.4.6. With a larger feasible solution range $[\beta, 1)$, CLUR can learn different uncertainty scores for different samples. This is because if all predictions $\widehat{\mathbf{y}}_Q$ are always close to one-hot labels, the uncertainty scores (e.g., reciprocal of winning scores in Sec. 3.4.5) of the predictions are almost the same. Second, we design a contrastive loss L_{CT_a} for a pair of augmented samples with an equal uncertainty relation,

$$L_{CT_a} = D[\widehat{\mathbf{y}}_{Q_1}, o(\widetilde{\mathbf{y}}_{Q_2})] + D[\widehat{\mathbf{y}}_{Q_2}, o(\widetilde{\mathbf{y}}_{Q_1})] \tag{3.9}$$

where D is cosine similarity and o is the detach operation to stop the gradient, similar to Eq. 3.3 in SimSiam. Eq. 3.9 scales between $\widetilde{\mathbf{y}}$ and $\widehat{\mathbf{y}}$, an intersection comparison. The total loss of module (a) is $L_{SUM_a} = L_{RC} + \gamma L_{CT_a}$, where γ is a constant.

Loss module (b). This performs best among the four loss choices in UEFTC by ϕ_1 and ϕ_2 in case 3. It also has two components, the same L_{RC} as Eq. 3.8, and its contrastive loss L_{CT_b}. Though the L_{CT_a} has the interaction comparison between projection $\widetilde{\mathbf{y}}_{Q_1}$ and prediction $\widehat{\mathbf{y}}_{Q_1}$, it is hard to explain why we should compare these two, and its effect is shown to be inconsistent by our experiments (Sec. 3.5.2). As a result, we only compare the uncertainty relations among the predictions $\widehat{\mathbf{y}}_Q$ in (b). For a pair of augmented samples with an equal uncertainty relation (case 1), we have its L_{CT_b} as,

$$L_{CT_b} = D[\widehat{\mathbf{y}}_{Q_1}, o(\widehat{\mathbf{y}}_{Q_2})] + D[\widehat{\mathbf{y}}_{Q_2}, o(\widehat{\mathbf{y}}_{Q_1})] \tag{3.10}$$

For the augmentations with an unequal uncertainty relation (case 2 or 3), different from Eq. 3.3 in SimSiam, we propose its L_{CT_b} as,

$$\begin{aligned} L_{CT_b} &= max\{[H(\widehat{\mathbf{y}}_{Q_1}) - H(o(\widehat{\mathbf{y}}_{Q_2}))] \times (\phi_2 - \phi_1), 0\} \\ &+ max\{[H(\widehat{\mathbf{y}}_{Q_2}) - H(o(\widehat{\mathbf{y}}_{Q_1}))] \times (\phi_1 - \phi_2), 0\} \end{aligned} \tag{3.11}$$

where $H(\widehat{\mathbf{y}}) = -\sum \widehat{y}_i \log(\widehat{y}_i)$ calculates entropy. A larger entropy means more uncertainty. In the first item of Eq. 3.11, the $(\phi_2 - \phi_1)$ calculates our pseudo unequal relations ($>$, $<$). The $[H(\widehat{\mathbf{y}}_{Q_1}) - H(o(\widetilde{\mathbf{y}}_{Q_2})))]$ calculates the model predicted unequal relations. If the predicted unequal relations and our pseudo unequal relations were the same, the predicted uncertainty scores from CLUR would be adaptive to our pseudo unequal relations, and the loss would be a constant 0 with no penalty. But if the predicted unequal relations and our pseudo unequal relations were different, the predicted uncertainty scores from CLUR would not be adaptive to our pseudo unequal relations, and there would be a positive loss as a penalty. The total loss is $L_{SUM_b} = L_{RC} + \gamma L_{CT_b}$.

Loss modules (c) & (d). Compared with (b), (c) shows the effectiveness of detach in CLUR, where L_{CT_c} only removes detach operation o; (d) shows the performance of predictors in CLUR by removing the predictors and learning via projections. Their loss functions are shown in the Sec. 3.4.6.

3.4.5 CLUR Inference: Uncertainty Score

During the testing process, its support and query samples from a testing episode all go through the first submodel by skipping the augmentation. Thus, our testing process is not affected by the augmentation, but only uses the knowledge of learning text classification and uncertainty estimation that is learned in the training process. Then, like [58, 126], we calculate uncertainty score Γ by the reciprocal of maximum probability in $\widehat{\mathbf{y}}_{Q_1}$, that is $\Gamma = \frac{1}{\max(\widehat{\mathbf{y}}_{Q_1})}$.

Generalization of CLUR. When applying uncertainty relations to other few-shot models, the two modules introduced in Sec. 3.4.3 are replaceable by other few-shot models to get their respective sample embeddings $\widetilde{\mathbf{X}}$ and query sample projects $\widetilde{\mathbf{Y}}_Q$. Then, the data augmentation for uncertainty relations (Sec. 3.4.2) and their respective loss functions (Sec. 3.4.4) are still applicable. We verify its generalization in Sec. 3.5.3.

3.4.6 Analysis of Model

Analysis Of Case 3 in Sec. 3.4.2

Assumption: We assume that the text semantics is not related to the rank of the words, but only related to the numbers of different words in a text (same assumption as bag-of-words (BOW) model [45]).

Conclusion: On above assumption, though each token contributes differently to text semantics, a larger difference (τ) in the numbers of mask tokens leads to a more accurate pseudo uncertainty relation.

Analysis: We token-mask an n-word text for twice. As a result, one augmented text has e_0+e_1 remaining words, the other one has e_0+e_2 remaining words after token-mask. Among the two augmented texts, e_0 words are the commonly remaining words in two texts after token-mask. The e_1 words and e_2 words are the two groups of uniquely remaining words for each augmented text, we assume $e_1 > e_2$. We set the semantic contributions of each word to a text for e_0 words as $\zeta_1, \zeta_2, ..., \zeta_{e_0}$, also set the semantic contributions of each word to a text for e_1 words and e_2 words as $\xi_1, \xi_2, ..., \xi_{e_1}$ and $\rho_1, \rho_2, ..., \rho_{e_2}$, respectively. Thus, based on BOW assumption, we have the semantics Ψ_1 and Ψ_2 of two augmented texts as,

$$\begin{aligned} \Psi_1 &= \sum_{i=1}^{e_0} \zeta_i + \sum_{j=1}^{e_1} \xi_j \\ \Psi_2 &= \sum_{i=1}^{e_0} \zeta_i + \sum_{j=1}^{e_2} \rho_j \end{aligned} \tag{3.12}$$

where each $\zeta_i \geq 0$, $\xi_j \geq 0$ and $\rho_j \geq 0$. Plus, each ζ_i, ξ_j and ρ_j in Eq. 3.12 is independent in BOW assumption. Due to the independence of each ζ_i, ξ_j, and ρ_j, the "each token

contributes differently to text semantics" in our conclusion has been satisfied. Then, in the current situation (case 2 with no margin), due to $e_1 > e_2$, we assume $\Psi_1 > \Psi_2$, which means pseudo uncertainty scores of the first sample should be smaller than the pseudo uncertainty scores of the second sample. But the assumption might be wrong, because we have no idea whether $\sum_{j=1}^{e_1} \xi_j > \sum_{j=1}^{e_2} \rho_j$ or not. If $\sum_{j=1}^{e_1} \xi_j > \sum_{j=1}^{e_2} \rho_j$ is true, then our pseudo uncertainty relation is accurate. Thus, we define $P(\Psi_1 > \Psi_2)$ to represent a probability that our pseudo relation is accurate for a pair of augmented text, given $e_1 > e_2$.

In case 3, due to the additional margin τ, the difference in numbers of remaining words is enlarged. Thus, we token-mask less $\nu = \llcorner n \times \tau \lrcorner$ words for the first augmentation. As a result, the number of remaining words of the first augmentation is changed from $e_0 + e_1$ to $e_0 + e_1 + \nu$, but the number of remaining words of the second augmentation is still $e_0 + e_2$. Thus, the semantics $\Psi_1^{'}$ of the first augmented text with τ is,

$$\Psi_1^{'} = \sum_{i=1}^{e_0} \zeta_i + \sum_{j=1}^{e_1} \xi_j + \sum_{k=1}^{\nu} \xi_k^{'} \tag{3.13}$$

where each $\xi_k^{'} \geq 0$ is the semantic contributions of each word to a text for ν additionally remaining words. As a result, to compare the accuracy of pseudo uncertainty relation for a pair of augmented texts with/without τ, we have the below by plugging in Eq. 3.12 and Eq. 3.13,

$$\begin{aligned} & P(\Psi_1^{'} > \Psi_2) - P(\Psi_1 > \Psi_2) \\ & = (\Psi_1^{'} - \Psi_2) - (\Psi_1 - \Psi_2) \\ & = \Psi_1^{'} - \Psi_1 \geq 0 \end{aligned} \tag{3.14}$$

Thus, we show that the probability that the pseudo uncertainty relation with τ is true, is higher than that without τ. In other words, the accuracy of pseudo uncertainty relation with τ is higher.

Loss Modules

Explain β in our revised cross-entropy loss L_{RC} of Eq. 3.8 in Sec. 3.4.4. Given a prediction $\widehat{\mathbf{y}}_Q \in \mathbb{R}^k$ and its respective ground truth $\mathbf{y}_Q \in \mathbb{R}^k$, we analyze the the relation between $L_{CE}(\widehat{\mathbf{y}}_Q, \mathbf{y}_Q) + log(\beta)$ and 0, where $\beta \in [0.5, 1)$. We detail the L_{CE} as below,

$$L_{CE}(\widehat{\mathbf{y}}_Q, \mathbf{y}_Q) = -\sum_{i=1}^{k} \mathbf{y}_Q^i log(\widehat{\mathbf{y}}_Q^i) \tag{3.15}$$

where $\mathbf{y}_Q^i$ and $\widehat{\mathbf{y}}_Q^i$ are the i-th entry of $\mathbf{y}_Q$ and $\widehat{\mathbf{y}}_Q$ respectively. Since $\mathbf{y}_Q$ is a one-hot vector and we assume its $\mathbf{y}_Q^j = 1$ and the rest entries of $\mathbf{y}_Q$ are all 0, we have below,

$$\begin{aligned} L_{CE}(\widehat{\mathbf{y}}_Q, \mathbf{y}_Q) &= -\mathbf{y}_Q^j log(\widehat{\mathbf{y}}_Q^j) \\ &= -log(\widehat{\mathbf{y}}_Q^j) \end{aligned} \tag{3.16}$$

where $L_{CE} > 0$. This is because $\widehat{\mathbf{y}}_Q$ is input to a softmax function in the PyTorch implementation of cross-entropy loss [6], and $\widehat{\mathbf{y}}_Q^j \in (0, 1)$. However, we do not expect the $\widehat{\mathbf{y}}_Q^j$ is always close 1, as it is now not 100% confidence belonging to j-th class due to data augmentation. Thus, we add β to L_{CE} as below,

$$L_{CE}(\widehat{\mathbf{y}}_Q, \mathbf{y}_Q) + log(\beta) = -log(\widehat{\mathbf{y}}_Q^j) + log(\beta) = log(\frac{\beta}{\widehat{\mathbf{y}}_Q^j}) \tag{3.17}$$

Then, we substitute Eq. 3.17 into Eq. 3.8, we have below,

$$L_{RC} = max[log(\frac{\beta}{\widehat{\mathbf{y}}_{Q_1}^j}), 0] + max[log(\frac{\beta}{\widehat{\mathbf{y}}_{Q_2}^j}), 0] \tag{3.18}$$

where L_{RC} equals to constant 0 with no penality, if $\mathbf{y}_{Q_1}^j \geq \beta$ and $\mathbf{y}_{Q_2}^j \geq \beta$. Thus, they both have feasible solution $[\beta, 1)$ in L_{RC}.

Below is our remaining losses, besides those in Sec. 3.4.4.
Loss module (a). In loss module (a), for an unequal uncertainty relation (case 2 or 3 in Sec. 3.4.2), our L_{CT_a} is,

$$\begin{aligned} L_{CT_a} = &max\{[H(\widehat{\mathbf{y}}_{Q_1}) - H(o(\widetilde{\mathbf{y}}_{Q_2}))] \times (\phi_2 - \phi_1), \\ &0\} + max\{[H(\widehat{\mathbf{y}}_{Q_2}) - H(o(\widetilde{\mathbf{y}}_{Q_1}))] \times (\phi_1 - \phi_2), 0\} \end{aligned} \tag{3.19}$$

The two items in Eq. 3.19 can be explained in a similar way to Eq. 3.11.
Loss module (c). It is designed to verify the effectiveness of detach in UEFTC. It has the same L_{RC} as Eq. 3.8. We only consider case 3 for it because we found case 3 achieved the best performance among three cases of uncertainty relations, when we used Loss module (b) for the experiments. Its L_{CT_c} in unequal uncertainty relation is,

$$L_{CT_c} = max\{[H(\widehat{\mathbf{y}}_{Q_1}) - H(\widehat{\mathbf{y}}_{Q_2})] \times (\phi_2 - \phi_1), 0\} \tag{3.20}$$

because there is no detach o, so there is no more difference between the two items in Eq. 3.11. The total loss is $L_{SUM_c} = L_{RC} + \gamma L_{CT_c}$.
Loss module (d). It is designed to verify the effectiveness of predictor in UEFTC by removing the predictors. Its L_{RC_d} is conducted on the projections $\widetilde{\mathbf{y}}_{Q_1}$. Its L_{CT_d} in unequal uncertainty relation is,

$$\begin{aligned} L_{CT_d} = &max\{[H(\widetilde{\mathbf{y}}_{Q_1}) - H(o(\widetilde{\mathbf{y}}_{Q_2}))] \times (\phi_2 - \phi_1), 0\} \\ &+ max\{[H(\widetilde{\mathbf{y}}_{Q_2}) - H(o(\widetilde{\mathbf{y}}_{Q_1}))] \times (\phi_1 - \phi_2), 0\} \end{aligned} \tag{3.21}$$

The total loss is $L_{SUM_d} = L_{RC_d} + \gamma L_{CT_d}$.

Table 3.1: Results of baselines and CLUR using fastText word embedding in 5-way 1-shot setting, where standard deviations are behind "±". More results are in Tab. 3.3.

Methods	Uncertainty Ratio (F1 Score, Eliminated Ratio)↑					AUROC ↑	AUPR↑
	0%	10%	20%	30%	40%		
20News in the 5-way 1-shot setting							
FTC-DS	47.56±1.56	55.76±1.38	62.92±1.25	69.86±1.11	75.77±1.04	68.17±2.15	68.20±1.29
DE	52.32±1.70	59.45±1.59	65.71±1.47	72.12±1.32	77.57±1.27	67.69±2.44	69.38±1.57
DE+Metric	52.33±1.61	59.63±1.44	65.73±1.36	72.04±1.26	77.61±1.15	68.02±2.38	69.44±1.45
MSD1	53.11±1.60	60.47±1.47	66.61±1.36	72.87±1.26	78.38±1.09	68.40±2.35	70.01±1.36
MSD2	52.54±1.32	60.09±1.19	66.54±1.10	72.59±1.04	77.96±0.93	68.49±1.91	69.78±1.01
SimSiam(CLUR-a-1)	53.30±1.57	60.63±1.43	66.86±1.32	73.19±1.23	78.59±1.16	68.74±2.29	70.89±1.36
CLUR-b-3	**54.53**±1.50	**62.06**±1.37	**68.29**±1.25	**74.59**±1.11	**80.02**±0.98	**70.50**±2.13	**73.71**±1.22
RCV1 in the 5-way 1-shot setting							
FTC-DS	51.32±1.64	59.71±1.49	66.16±1.33	72.83±1.23	78.65±1.12	70.48±2.32	73.99±1.22
DE	55.42±1.62	62.96±1.50	68.91±1.37	74.99±1.22	80.09±1.14	70.72±2.34	75.12±1.12
DE+Metric	54.89±1.68	62.50±1.52	68.41±1.34	74.59±1.25	79.78±1.20	70.61±2.46	74.51±1.24
MSD1	54.91±1.79	62.32±1.64	68.27±1.48	74.60±1.36	79.82±1.26	70.11±2.50	73.67±1.35
MSD2	55.54±1.65	62.96±1.50	68.91±1.39	75.18±1.30	80.39±1.17	71.12±2.37	75.34±1.23
SimSiam(CLUR-a-1)	54.12±1.97	61.66±1.79	67.98±1.67	74.47±1.49	79.71±1.38	71.10±2.73	74.24±1.56
CLUR-b-3	**55.89**±1.60	**63.48**±1.44	**69.47**±1.35	**75.62**±1.23	**80.91**±1.12	**72.31**±2.26	**77.00**±1.10
Amazon in the 5-way 1-shot setting							
FTC-DS	59.06±1.49	66.81±1.30	72.65±1.27	78.22±1.14	82.73±1.01	70.05±1.96	79.03±0.97
DE	59.87±1.94	66.91±1.79	72.60±1.65	78.25±1.49	83.10±1.38	70.34±2.62	78.48±1.62
DE+Metric	61.36±1.65	68.39±1.51	73.83±1.37	79.13±1.27	83.55±1.15	70.66±2.37	79.63±1.15
MSD1	61.30±1.74	68.08±1.60	73.60±1.47	78.99±1.36	83.48±1.24	70.00±2.55	78.41±1.38
MSD2	61.56±1.34	68.30±1.20	73.87±1.11	79.24±1.05	83.78±1.00	70.70±1.93	80.02±0.90
SimSiam(CLUR-a-1)	61.42±1.87	68.13±1.73	73.60±1.59	78.93±1.42	83.37±1.27	69.66±2.52	78.66±1.41
CLUR-b-3	**63.32**±1.38	**70.08**±1.26	**75.41**±1.17	**80.69**±1.04	**85.13**±0.92	**71.59**±1.93	**81.78**±0.89
HuffPost in the 5-way 1-shot setting							
FTC-DS	40.65±1.48	48.92±1.33	56.25±1.25	63.64±1.14	70.31±1.08	66.35±2.21	61.35±1.37
DE	42.46±1.63	50.15±1.51	56.87±1.40	64.02±1.35	70.34±1.24	64.72±2.50	58.82±1.72
DE+Metric	42.55±1.40	50.27±1.31	57.19±1.19	64.33±1.10	70.64±1.07	65.48±2.26	59.96±1.29
MSD1	43.25±1.23	50.91±1.16	57.64±1.09	64.63±1.03	70.70±0.99	65.09±1.98	60.59±1.18
MSD2	42.92±1.12	50.70±1.03	57.32±0.97	64.23±0.92	70.58±0.87	64.88±1.80	59.41±1.04
SimSiam(CLUR-a-1)	43.18±1.31	50.73±1.24	57.46±1.17	64.58±1.12	70.80±1.02	65.42±1.96	61.41±1.24
CLUR-b-3	**44.05**±1.62	**51.62**±1.48	**58.26**±1.38	**65.20**±1.27	**71.39**±1.17	**66.50**±2.43	**63.07**±1.53

3.5 Experiments

3.5.1 Experimental Setup

Datasets. We use four real-world datasets: (1) 20 Newsgroups (**20News**) [83] includes 20 news categories with 18,828 documents in it. (2) Amazon Reviews (**Amazon**) [110] is a set of reviews [60]. We use its subset provided by [6], which has 1000 reviews from each category. (3) HuffPost headlines (**HuffPost**) provides news headlines from HuffPost between 2012 and 2018 [114]. It has 36900 headlines with 41 classes. They are shorter and less grammatical than formal sentences. (4) **RCV1** collects Reuters articles from 1996 to 1997 [90]. We use its 71 second-level topics as labels, and discard its multi-label articles. Each dataset is split in the same way as [6].

[6]The PyTorch implementation of cross-entropy loss is "torch.nn.CrossEntropyLoss", which can be found from its official API.

Metrics. To evaluate the performance of UEFTC, we use three metrics. The first two are the area under the receiver operating characteristic curve (**AUROC**) and the area under the precision-recall curve (**AUPR**), which are broadly applied in uncertainty estimation [61, 63, 105, 195]. Higher AUROC and AUPR both mean a higher probability that a true prediction has a lower uncertainty score than a false prediction. Besides, to simulate the performance improvement of uncertainty scores with human involvement, we scale classification accuracy in different eliminated ratios [58,189]. Concretely, for a testing episode with N query samples and eliminated ratio r, the most uncertain predictions in size of $N \times r$ are set as true. The more accurate the uncertainty scores we obtain, the more misclassified predictions will be set as true predictions under the same r, resulting in a larger F1 score. The F1 score under 0% eliminated ratio is the model classification performance.

Baselines and Ablation Settings. We use six baselines. **FTC-DS** [6] is the SOTA few-shot text classification model described in Sec. 3.3.2. To ensure fairness, all other baselines are also built on FTC-DS, like CLUR. [58, 189] have the same tasks as ours, but they use numerous training samples. [189] propose two methods: Dropout-Entropy (**DE**) is a dropout-based model. **DE+Metric** additionally uses metric learning. There are two other methods in [58] applicable to UEFTC: **MSD1** uses mix-up to manually set pseudo uncertainty scores; **MSD2** adds self-ensembling components to MSD1. We refer to our CLUR with loss module (a) using equal uncertainty relation as **SimSiam** [16], since it uses similar key structures and key loss L_{CT_a} (Eq. 3.9) as SimSiam (Eq. 3.3).

For the ablation studies, we design five comparisons listed in Tab. 3.4. They are in different cases and structures to compare different designs: detach (**DT**), predictor (**PD**), and intersection comparison (**IT**) between the projection and prediction (described in Sec. 3.3.3). We use CLUR-{$\cdot$}-{$\star$} to represent a CLUR using the loss module {$\cdot$} in case {$\star$} described in Sec. 3.4.4 and 3.4.2 respectively.

Implementation Details. We use fastText [70] as the word embedding for our experiments by default. Besides fastText, we also test BERT [21] word embedding for 5-way 1-shot on 20News. Our parameter settings are listed in Sec. 3.5.3 and Tab. 3.7.

Table 3.2: 5-way 1-shot using BERT word embedding on 20News.

ID	Methods	Classification F1 Score ($r = 0\%$)↑	AUROC↑	AUPR↑
1	FTC-DS	38.92	64.54	58.58
2	DE	44.81	63.40	59.45
3	DE+Metric	45.16	63.66	59.84
4	MSD1	45.75	64.00	61.09
5	MSD2	45.47	63.18	59.19
6	SimSiam(CLUR-a-1)	44.40	63.82	59.30
7	CLUR-b-3	**46.54**	**64.78**	**62.57**

Table 3.3: Results of baselines and CLUR using fastText word embedding in 5-way 5-shot setting, where standard deviations are behind "±". More results are in Tab. 3.1.

Methods	Uncertainty Ratio (F1 Score, Eliminated Ratio)↑					AUROC ↑	AUPR↑
	0%	10%	20%	30%	40%		
20News in the 5-way 5-shot setting							
FTC-DS	63.83±1.18	70.87±1.05	76.76±0.92	82.31±0.79	86.86±0.68	76.22±1.69	85.46±0.54
DE	65.95±1.27	72.56±1.08	77.62±1.00	82.56±0.93	86.69±0.88	74.15±2.09	84.10±0.84
DE+Metric	65.82±1.07	72.58±0.95	77.81±0.83	83.08±0.77	87.34±0.70	75.52±1.72	84.65±0.84
MSD1	65.97±1.20	72.61±1.04	77.88±0.93	82.93±0.87	87.09±0.86	73.98±2.00	83.15±1.04
MSD2	65.83±1.06	72.35±0.95	77.83±0.85	83.10±0.79	87.24±0.74	74.72±1.71	83.37±0.79
SimSiam(CLUR-a-1)	66.02±1.31	72.71±1.17	78.02±1.01	83.10±0.94	87.15±0.90	74.78±2.03	84.10±0.96
CLUR-b-3	**66.88**±1.16	**73.71**±0.99	**79.20**±0.85	**84.26**±0.78	**88.21**±0.76	**76.50**±1.77	**86.39**±0.57
RCV1 in the 5-way 5-shot setting							
FTC-DS	72.28±1.63	78.82±1.42	83.35±1.23	87.79±1.11	91.29±0.97	77.26±2.54	89.65±0.75
DE	73.13±1.51	79.48±1.25	84.64±1.10	89.18±0.97	92.16±0.90	**80.11**±2.39	91.34±0.65
DE+Metric	74.52±1.46	80.49±1.19	84.90±1.11	89.01±1.06	92.20±0.93	77.57±2.56	90.86±0.69
MSD1	74.05±1.55	80.51±1.35	85.42±1.16	89.76±1.11	92.65±0.99	79.53±2.46	90.98±0.78
MSD2	74.44±1.25	80.75±1.03	85.22±0.98	89.52±0.86	92.56±0.75	79.24±2.01	91.2±0.57
SimSiam(CLUR-a-1)	73.51±1.49	80.14±1.27	84.60±1.10	88.76±1.02	92.08±0.91	79.11±2.37	90.80±0.71
CLUR-b-3	**75.88**±1.37	**82.09**±1.23	**86.31**±1.11	**90.28**±0.98	**93.17**±0.83	79.65±2.29	**91.55**±0.65
Amazon in the 5-way 5-shot setting							
FTC-DS	81.23±1.05	86.75±0.83	90.62±0.70	93.67±0.61	95.81±0.54	81.00±1.75	94.66±0.32
DE	81.07±1.26	86.48±1.05	90.31±0.93	93.49±0.77	95.58±0.66	80.93±2.15	94.29±0.51
DE+Metric	81.05±1.23	86.54±1.04	90.33±0.89	93.36±0.77	95.46±0.67	80.72±2.14	94.17±0.52
MSD1	81.79±1.24	87.01±1.04	90.81±0.85	93.84±0.75	95.91±0.63	81.07±2.05	94.72±0.47
MSD2	81.06±1.09	86.44±0.91	90.20±0.79	93.24±0.69	95.36±0.62	80.12±1.90	94.08±0.44
SimSiam(CLUR-a-1)	80.75±1.33	86.26±1.16	90.02±0.99	92.98±0.83	95.09±0.75	79.73±2.24	93.66±0.56
CLUR-b-3	**81.95**±1.09	**87.37**±0.90	**91.49**±0.76	**94.47**±0.57	**96.21**±0.51	**82.35**±1.79	**95.16**±0.36
HuffPost in the 5-way 5-shot setting							
FTC-DS	62.28±0.92	69.44±0.87	75.70±0.76	81.60±0.69	86.23±0.63	75.82±1.29	84.06±0.52
DE	63.80±1.20	70.79±1.06	76.48±0.99	81.86±0.91	86.22±0.79	74.74±1.74	83.50±0.72
DE+Metric	63.58±1.27	70.45±1.14	76.31±1.03	81.75±0.86	86.01±0.84	74.72±1.79	83.42±0.79
MSD1	**64.11**±1.14	**71.16**±1.03	**76.83**±0.92	**82.09**±0.85	86.35±0.77	74.80±1.64	83.64±0.76
MSD2	63.58±0.98	70.43±0.88	76.19±0.82	81.51±0.78	85.90±0.71	74.28±1.44	83.16±0.60
SimSiam(CLUR-a-1)	63.67±1.29	70.39±1.15	75.87±1.07	81.25±0.96	85.71±0.89	73.74±1.84	82.87±0.82
CLUR-b-3	63.55±1.37	70.74±1.22	76.63±1.12	82.04±1.00	**86.48**±0.89	**75.74**±1.89	**84.10**±0.82

Table 3.4: Ablation study of CLUR using fastText word embedding in the 5-way 5-shot setting on Amazon dataset, where standard deviations are behind "±." The "DT" means detach, "IT" means intersection, and "PD" means predicotr.

Methods	DT	IT	PD	Uncertainty Ratio (F1 Score, Eliminated Ratio) ↑					AUROC ↑	AUPR ↑
				0%	10%	20%	30%	40%		
Amazon in the 5-way 5-shot setting										
CLUR-b-3	✓	×	✓	**81.95**±1.09	**87.37**±0.90	**91.49**±0.76	**94.47**±0.57	**96.21**±0.51	**82.35**±1.79	**95.16**±0.36
CLUR-c-3	×	×	✓	81.44±1.09	86.91±0.94	90.59±0.77	93.63±0.70	95.76±0.61	81.26±1.92	94.52±0.43
CLUR-d-3	✓	×	×	80.17±2.09	85.90±1.76	89.93±1.48	93.33±1.23	95.58±1.02	81.13±3.05	94.33±0.92
CLUR-a-2	✓	✓	✓	80.83±1.29	86.32±1.12	90.14±0.96	93.33±0.82	95.50±0.71	80.69±2.15	94.23±0.55
CLUR-b-2	✓	×	✓	80.59±1.23	86.11±1.06	90.00±0.91	93.25±0.80	95.42±0.70	80.79±2.07	94.17±0.52
CLUR-c-2	×	×	✓	80.90±1.19	86.31±1.01	90.05±0.84	93.08±0.75	95.20±0.66	80.11±2.05	93.91±0.48

3.5.2 Experimental Results

Comparison With Baselines

Tab. 3.1 and 3.3 report the CLUR improvement in UEFTC using fastText in the 5-way 1-shot and 5-way 5-shot settings respectively. We repeat the testing process 30 times with the same dropout rate. And we calculate the mean and standard deviation for each metric, which are reported in the tables. From the two tables, we discuss below questions:
1. Are the learned pseudo uncertainty scores from uncertainty relations better than manual setting ones? Yes, CLUR-b in case 3 performs better than MSD1 and MSD2, which both manually set pseudo uncertainty scores, such as 4.52% AUPR improvement than MSD1 in the 5-way 1-shot setting on RCV1 in Tab. 3.1, and 1.97% AUROC improvement than MSD2 in the 5-way 5-shot setting on 20News in Tab. 3.3. Concretely, MSD1 and MSD2 both use the mix-up to augment the texts and then manually set the mix-up coefficient as pseudo uncertainty scores. Compared with them, CLUR learns the pseudo uncertainty scores by Eq. 3.11, instead of the manual setting. Though MSD1 has higher F1 scores in the eliminated ratios (0%-30%) than CLUR in 5-way 5-shot on HuffPost, CLUR surpasses MSD1 in AUROC and AUPR in the same setting. It means that CLUR predicts more accurate uncertainty scores in total. Thus, learned pseudo uncertainty scores from uncertainty relations are more accurate than manual setting ones.
2. Is CLUR better than traditional uncertainty estimation methods (dropout, metric learning, self-ensemble, and pseudo-label-based methods) applicable to a few training samples? Yes, they are. In detail, DE uses dropout, and DE+Metrics additionally uses metric learning to reduce uncertainty. MSD1 manually sets pseudo uncertainty scores; MSD2 extra uses self-ensemble [129] to reduce uncertainty. In the two tables, CLUR beats them by its uncertainty relations and structure design in vast comparisons. For example, CLUR improves 4.39% and 4.08% F1 scores with a 10% eliminated ratio than DE and DE+Metric, respectively, in the 5-way 1-shot setting on 20News (Tab. 3.1). Therefore, CLUR beats traditional uncertainty estimation methods applicable to few-shot settings.

Below, we discuss the results using BERT embedding for 5-way 1-shot on 20News, shown in Tab. 3.2.
3. Is CLUR effective on BERT embeddings? Yes, our unequal uncertainty relation in case 3 and (b) loss module in CLUR perform better on 20News, using BERT embeddings. For example, CLUR-b-3 improves 2.42% AUPR than MSD1 in Tab. 3.2.

Ablation Study

Tab. 3.4 shows ablation study results. From the results, we conclude as below.
4. Which combo choices of loss modules and uncertainty relations perform better? CLUR using detach, predictor, no intersection comparison (L_{CT_b} in Eq. 3.11) in an uncertainty relation with a margin (case 3 on Sec. 3.4.2) performs better on almost all

datasets. For example, CLUR-b-3 improves 2.80% AUROC than CLUR-c-2 in a 5-way 5-shot setting on Amazon (Tab. 3.4).

5. Among the three cases of uncertainty relations, which one performs better? Case 3 performs the best among the three cases. In detail, from the view of L_{CT_b} in Eq. 3.11, we compare CLUR-b-3 and CLUR-b-2. We see that CLUR-b-3 is better than CLUR-b-2. Thus, using a margin (case 3) performs better than without a margin (case 2). In addition, from the view of L_{CT_a}, we compare CLUR-a-2 and CLUR-a-1 (SimSiam). Since CLUR-a-2 performs better than CLUR-a-1, we conclude that case 2 is better than case 1. Together, case 3 performs the best among the three cases.

6. Do detach, intersection and predictor (introduced in Sec. 3.3.3) improve uncertainty estimation? From Tab. 3.4, we can conclude below. (i) The detach is effective by comparing the CLUR-b-3 and CLUR-c-3. This is because the detach acts as a form of structural ensemble and helps reduce uncertainty. (ii) As for the intersection comparison between projections and predictions, it is inconsistently effective in UEFTC by finding the slight difference between CLUR-a-2 and CLUR-b-2. (iii) The predictor leads to the most obvious improvement by comparing CLUR-b-3 and CLUR-d-3. This is because the predictor provides more parameters and better handles classification and uncertainty estimation simultaneously.

Generalization to other few-shot models. We conducted a generalization analysis, as shown in Tab. 3.5 and discussed in Sec. 3.5.3.

Experiment on a high-risk domain dataset. Besides the four commonly used datasets, we are also interested in exploring the effectiveness of CLUR on public high-risk datasets, such as healthcare. Therefore, we conducted experiments on a medical-domain dataset, as shown in Tab. 3.6 and discussed in Sec. 3.5.3.

Table 3.5: The comparison between baselines and our CLUR-b-3 in setting CNN as embeddings and Prototypical Network as classifiers on 20News with the 5-way 1-shot setting.

Methods	Uncertainty Ratio (F1 Score, Eliminated Ratio)↑					AUROC ↑	AUPR↑
	0%	10%	20%	30%	40%		
FTC-DS	27.12±3.58	35.75±3.43	43.48±3.29	51.64±3.09	58.96±2.95	55.75±5.96	37.11±6.91
DE	29.83±3.52	38.09±3.36	45.55±3.18	53.51±3.00	60.72±2.88	58.81±5.70	41.14±6.81
DE+Metric	31.09±3.04	39.22±2.89	46.54±2.76	54.35±2.56	61.35±2.41	58.76±4.74	42.17±5.05
MSD1	30.96±2.84	39.06±2.68	46.36±2.58	54.08±2.48	61.04±2.38	57.75±4.76	40.13±4.33
MSD2	30.36±3.53	38.44±3.34	45.71±3.17	53.53±2.99	60.60±2.77	57.72±5.26	40.54±5.85
SimSiam(CLUR-a-1)	30.39±3.42	38.52±3.28	45.81±3.14	53.55±2.97	60.66±2.76	57.58±5.32	40.62±5.90
CLUR-b-3	**31.77**±3.32	**40.16**±3.09	**47.54**±2.92	**55.37**±2.73	**62.47**±2.56	**59.20**±5.18	**43.89**±5.75

3.5.3 More about Experiments

Implementation Details.

We use fastText [70] as the word embedding by default. For all experiments, we set $\beta = 0.75$. We list the parameter settings on Tab. 3.7, which are parameters used to get our reported

Table 3.6: Comparing baselines and CLUR-b-3 using FTC-DS on the Med-Domain dataset with the 5-way 1-shot setting.

Methods	Uncertainty Ratio (F1 Score, Eliminated Ratio)↑					AUROC ↑	AUPR↑
	0%	10%	20%	30%	40%		
FTC-DS	50.63±1.79	58.98±1.55	65.63±1.40	71.69±1.28	77.08±1.23	67.42±2.37	70.24±1.66
DE	56.01±1.83	63.13±1.67	69.36±1.53	75.17±1.44	80.36±1.32	70.94±2.54	75.53±1.43
DE+Metric	54.98±2.12	62.06±1.96	68.32±1.85	74.31±1.71	79.80±1.55	71.01±2.89	75.62±1.79
MSD1	55.93±1.99	62.88±1.82	69.04±1.70	74.85±1.60	80.02±1.44	70.10±2.71	74.39±1.65
MSD2	55.99±1.50	62.96±1.39	69.04±1.32	74.78±1.21	79.94±1.08	70.15±2.10	75.82±1.08
SimSiam(CLUR-a-1)	54.48±1.69	61.49±1.62	67.78±1.51	73.89±1.39	79.43±1.32	70.64±2.36	74.31±1.49
CLUR-b-3	**56.81**±1.69	**63.87**±1.51	**70.16**±1.42	**76.10**±1.32	**81.44**±1.21	**72.31**±2.36	**77.29**±1.31

Table 3.7: Parameter settings that we use to get our CLUR-b-3 results with fastText embeddings.

Datasets	5-way 1-shot				5-way 5-shot			
	γ	Φ_1	τ	Φ_2	γ	Φ_1	τ	Φ_2
20News	0.1	0.1	0.1	0.3	1	0.1	0.1	0.3
RCV1	1	0.1	0.1	0.3	1	0.1	0.05	0.25
Amazon	1	0.1	0.05	0.25	1	0.1	0.15	0.35
HuffPost	1	0.15	0.1	0.4	1	0.15	0.1	0.4

CLUR-b-3 results. For the 5-way 1-shot using BERT embedding on 20News, we set $\gamma = 0.1$, $\Phi_1 = 0.1$, $\tau = 0.1$, and $\Phi_2 = 0.3$. The ablation studies also use the respective parameters.

Generalization On Other Few-Shot Model

Our CLUR-b-3 is experimentally effective on CNN embedding and Prototypical Network classifier. We conducted experiments using CNN embeddings [189] and Prototypical Network [2] classifiers for the UEFTC, which are common used in few-shot learning. We compared our CLUR-b-3 model to the baselines in the 5-way 1-shot setting on the 20News dataset. The results of our experiments are presented in Tab. 3.5. Our results indicate that our CLUR-b-3 model surpasses all the baselines, such as achieving over 3.35 points AURP compared to MSD2 in Tab. 3.5. These results suggest that our CLUR-b-3 model is effective not only for FTC-DS [1] but also for CNN embeddings and Prototypical Network classifiers. As a result, CLUR exhibits potential for generalization to other few-shot models.

Experiment On A Med-Domain Dataset

Besides the four common-used datasets, we also explore the effectiveness of CLUR on public high-risk datasets like healthcare. We choose Med-Domain dataset [19], which has medical transcription and labels. We use its subset of 1294 samples among 29 classes for our experiments, where the subset is released in our data split. We compare the performance between

baselines and our CLUR-b-3 based on FTC-DS, shown as Tab. 3.6. We conclude that our CLUR still works well in the high-risk domain, say, the healthcare domain. This is because our CLUR surpasses all baselines, such as improving 2.16 points AUROC over MSD2 in Tab. 3.6.

Chapter 4

Uncertainty Estimation on Sequential Labeling[1]

Sequential labeling is a task predicting labels for each token in a sequence, such as Named Entity Recognition (NER). NER tasks aim to extract entities and predict their labels given a text, which is important in information extraction. Although previous works have shown great progress in improving NER performance, uncertainty estimation on NER (UE-NER) is still underexplored but essential. This work focuses on UE-NER, which aims to estimate uncertainty scores for the NER predictions. Previous uncertainty estimation models often overlook two unique characteristics of NER: the connection between entities (i.e., one entity embedding is learned based on the other ones) and wrong span cases in the entity extraction subtask. Therefore, we propose a Sequential Labeling Posterior Network (SLPN) to estimate uncertainty scores for the extracted entities, considering uncertainty transmitted from other tokens. Moreover, we have defined an evaluation strategy to address the specificity of wrong-span cases. Our SLPN has achieved significant improvements on three datasets, such as a 5.54-point improvement in AUPR on the MIT-Restaurant dataset. Our code is available at `https://github.com/he159ok/UncSeqLabeling_SLPN`.

4.1 Introduction

Named entity recognition (NER) is a popular task in the information extraction domain [82], which involves two steps, detecting entity spans and predicting the entity labels. In many information extraction scenarios, there are significant consequences for relying on inaccurate NER predictions. For example, extracting an inaccurate time can lead to erroneous policy analysis, or misclassifying a person's name for a time can result in a privacy breach. There-

[1]J. He, Y. Lin, S. Lei, C. Lu, F. Chen. Uncertainty Estimation on Sequential Labeling via Uncertainty Transmission. ArXiv preprint arXiv:2311.08726 (2023).

fore, it is crucial to determine whether we can trust the NER predictions or not. As a result, our goal is to enhance Uncertainty Estimation in NER (UE-NER), which aims to quantify prediction confidence in NER tasks.

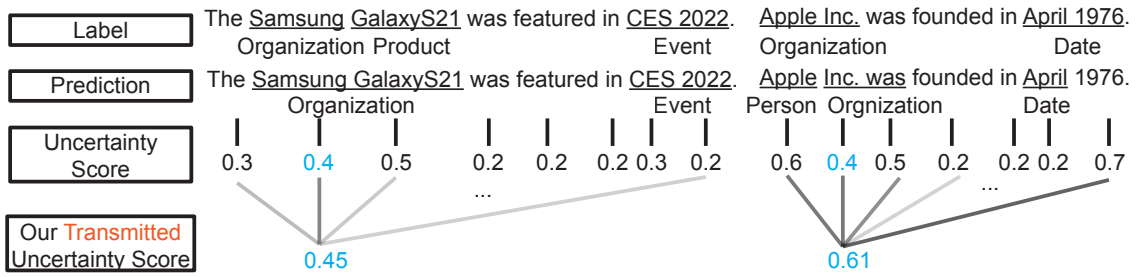

Figure 4.1: In this example, though the tokens "Samsung" and "Inc." both have the same uncertainty score of 0.4, the context in the right case exhibits higher uncertainty. This suggests that "Inc." should be considered more uncertain than "Samsung." Therefore, we propose transmitting the predicted uncertainty from other tokens to a given token.

The NER task differs from general classification (e.g., text classification [113] in two key ways, making previous uncertainty estimation models suboptimal for UE-NER.

First, the predicted entity labels in the NER task are directly dependent on the token embeddings, and uncertainty transmission between token embeddings is unique in NER. Concretely, given an example text "Barack Obama was born in Honolulu, Hawaii," the entity label "person" applies to "Barack Obama." The embedding of the token "Obama" is obtained by accumulating its own embedding and embeddings from other tokens in Recurrent Neural Network [111] and transformer [155]. Consequently, if a token embedding has higher uncertainty, the other token embedding will have more transmitted uncertainty from the token. Since token embeddings directly affect token labels and further affect entity labels, high uncertainty in a token embedding will result in a predicted entity label with high uncertainty. Therefore, in the context of UE-NER, a token uncertainty in UE-NER consists the individual token uncertainty and the uncertainty transmitted from other tokens.

However, the current uncertainty estimation methods ignore the uncertainty transmission between tokens. Especially, current uncertainty estimation methods can be classified into two main categories: parameter-distribution-based methods, such as Bayesian Neural Networks (BNN) [104, 125], which learns a distribution over the model parameters; and sample-distribution-based methods, which calculate uncertainty scores based on the distribution of training samples [10, 58, 129]. These methods primarily focus on image or text classification, where correlations between different images or texts are weak or limited. Consequently, they overlook the uncertainty transmission inherent in sequential labeling. Since sequential labeling plays a pivotal role in Natural Language Processing (NLP), with NER as a representative example, it is imperative for us to address UE-NER by considering uncertainty transmission, shown as Figure 4.1.

The second characteristic of NER tasks is that they involve an additional step, entity ex-

traction, besides entity classification. In contrast to previous text classification tasks [113], which focus solely on sample classification, NER tasks require the additional task of extracting entity spans, such as locating "Barack Obama." However, entity span extraction may predict entities with wrong span (WS), such as predicting "Obama was" as an entity. These WS entities lack ground truth entity labels and evaluating uncertainty estimation requires ground truth labels, thus these entities cannot be used for evaluating uncertainty estimation. Therefore, we require an innovative approach to evaluate a UE-NER model that takes into account these WS entities.

To address the first issue, we propose a Sequential Labeling Posterior Network (SLPN) for transmitting uncertainty. This network is built upon an evidential neural network framework [10] with a novel design to transmit uncertainty from other tokens. For the second issue, we categorize the ground truth entities and predicted entities into three groups: unique entities in the ground truth, unique entities in the prediction, and shared entities between the ground truth and prediction. We, then, treat WS entity detection as a separate subtask, in addition to out-of-domain (OOD) detection, which is a common task used to evaluate uncertainty estimation [195]. The WS and OOD detections use different combinations of the three-group entities. Furthermore, we evaluate the performance of a UE-NER model by computing a weighted sum of WS entity detection and OOD detection performance, providing a comprehensive assessment of the UE-NER model. Our contributions are as follows.

- Since each token embedding is influenced by other tokens within a given text, and token embedding directly affects the uncertainty of predicted entity labels, we propose a novel method to transmit uncertainty between tokens using a revised self-attention. To the best of our knowledge, we are the first to consider uncertainty transmission in UE-NER.
- Because of the existence of WS entities in the NER task, we have found that traditional evaluation methods for uncertainty estimation are inapplicable in UE-NER. Therefore, we propose a novel uncertainty estimation evaluation to evaluate both OOD and WS detection tasks.

4.2 Related Work

Named Entity Recognition. Named Entity Recognition (NER) is a task focused on extracting and classifying entities within text. It serves as a prominent example of sequential labeling, where each token in a sequence is assigned a label. Various techniques have been employed for NER, including Recursive Neural Networks (e.g., LSTM [46]), pretrained transformer models (e.g., BERT [21]). In some cases, Conditional Random Fields (CRF) are incorporated into token encoders, such as LSTM+CRF [82], to enhance performance.

Further, recent experiments have explored the use of Large Language Models (LLMs) for

NER. An LLM-based approach treats NER as a generative task, with each turn generating one category of entities [162]. However, it is noticeable that [162] found that GPT3-based NER solutions [28] did not outperform pretrained transformer based method. Since both pretrained transformer-based methods and LLMs are built on transformer architectures [155] and pretrained transformer-based methods take NER as sequential generation rather than sequential labeling, as well as perform better than GPT-3 on the NER task, our research focuses on UE-NER using pretrained-transformer-based methods.

Uncertainty estimation on natural language processing. Generally, for the usage of uncertainty estimation on training data, the uncertainty score helps with sample selection in active learning [165]. For usage on the testing data, uncertainty estimation mainly serves two tasks: *OOD detection* [48], where the testing samples include OOD samples, and the task aims to identify these OOD samples; and *misclassified result detection*: where testing samples are in-domain [58, 63, 189]. Our work specifically focuses on OOD detection in the testing samples.

In the NER domain, [8, 98, 121] estimated uncertainty scores on unlabeled training data for active learning. [156] were the first to apply uncertainty estimation to address misclassification in NER testing data using techniques like dropout [30] and deterministic uncertainty estimation methods (e.g., Gaussian process [96]). Additionally, on the testing samples, [194] were the first to apply uncertainty estimation to detect OOD instances in NER testing data. Compared to [194], who assigned different weights to different tokens, our work focuses on the transmission of uncertainty from other tokens to a specific token.

4.3 UE-NER Task Setting

Before we introduce UE-NER, we first introduce NER tasks, which is a representative sequential labeling task. Given a text $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n]$ with n tokens, where $\mathbf{x}_i \in \mathbb{R}^{h \times 1}$ is an embedding of a token, NER task aims at learning a NER model predicting their token labels. Then, the entities are extracted by the token labels based on the BIOES mechanism [17] (e.g., "Brack" with B-PER label, and "Obama" with I-PER label). Moreover, the extracted entities are classified by merging the entity tokens. For example, "Brack Obama" is categorized as a Person because these two tokens are categorized as the beginning and intermediate of the person label.

For the UE-NER task, we aim to learn a UE-NER model Φ to predict the confidence of each predicted token label. We apply Φ for OOD detection, which is a common task to evaluate uncertainty estimation [195]. Concretely, the training data and validation data for Φ are the in-domain (ID) text without OOD entities. The testing data of Φ includes both ID text and OOD text, where OOD text has both ID and OOD entities. A better Φ should detect more OOD entities in the testing set and have better NER performance.

4.4 Preliminary: Posterior Network

The parameter-distribution-based uncertainty estimation method is usually implemented via ensemble sampling [30] and thus requires multiple forward passes to estimate uncertainty, which is time-consuming. In contrast, Evidential Deep Learning (EDL) [139] is a representative sample-distribution-based uncertainty estimation method and is implemented via a deterministic model, thus requiring only one forward pass to estimate uncertainty. Due to its efficiency, we choose EDL.

In EDL, considering the classification task and given the input vector $\mathbf{X}$, the class prediction $y \in [c]$ for an input sample follows a categorical distribution with c classes.The categorical distribution naturally follows a Dirichlet distribution, i.e.

$$y \sim \mathrm{Cat}(\mathbf{p}), \quad \mathbf{p} \sim \mathrm{Dir}(\boldsymbol{\alpha}) \tag{4.1}$$

The expected class probability $\bar{\mathbf{p}}$ is calculated as below,

$$\alpha_0 = \sum_{k=1}^{c} \alpha_k, \quad \bar{\mathbf{p}} = \frac{\boldsymbol{\alpha}}{\alpha_0} \tag{4.2}$$

where $\mathrm{Dir}(\boldsymbol{\alpha})$ is an approximation of the posterior distribution of class probabilities, conditioned on the input feature vector. The concentrate parameters $\boldsymbol{\alpha} = [\alpha_1, \alpha_2, ..., \alpha_c]$ can be interpreted as the evidence for the given example belonging to the corresponding class [71]. The evidence is the count of pseudo support from training samples.

As a representative model of EDL, Posterior Network (PN) [10] is originally designed for image classification and involves two main steps. First, a feature encoder maps the raw features into a low-dimensional latent space. Second, a normalizing flow such as Radial [135] is used to estimate class-wise density on the latent space, which is proportional to the class-wise evidence. Essentially, a greater density for a particular class implies stronger evidence belonging to this class for the given example.

PN is trained with the sum of two loss $\mathcal{L}^{\mathrm{UCE}}$ and $\mathcal{L}^{\mathrm{ER}}$ for N training samples as below,

$$\mathcal{L}^{\mathrm{UCE}} = \frac{1}{N} \sum_{i=1}^{N} \mathbb{E}_{\mathbf{p}_i \sim \mathrm{Dir}(\mathbf{p}_i|\boldsymbol{\alpha}_i)} [\mathrm{CE}(\mathbf{p}_i, y_i)] \tag{4.3}$$

$$\mathcal{L}^{\mathrm{ER}} = -\frac{1}{N} \sum_{i=1}^{N} \mathbb{H}(\mathrm{Dir}(\mathbf{p}_i|\boldsymbol{\alpha}_i)) \tag{4.4}$$

where the Uncertainty Cross Entropy (UCE) loss $\mathcal{L}^{\mathrm{UCE}}$ encourages high evidence for the ground-truth category and entropy regularization $\mathcal{L}^{\mathrm{ER}}$ encourages a smooth Dirichlet distribution.

4.5 Model

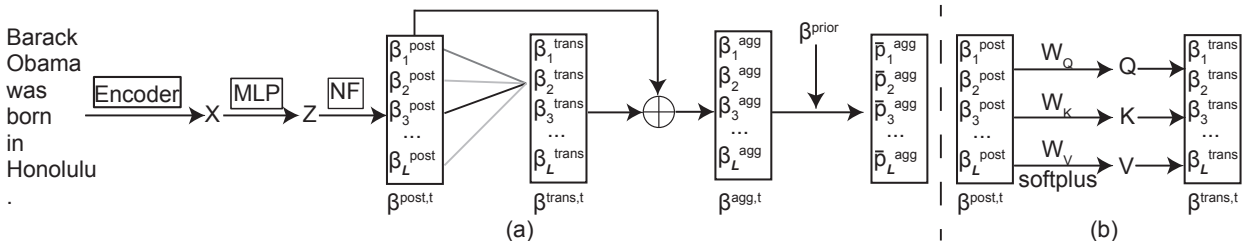

Figure 4.2: (a) A diagram of our SLPN model illustrates how we achieve uncertainty transmission through a revised self-attention mechanism applied to all tokens. Specifically, the SLPN model begins by generating a text embedding matrix $\mathbf{X}$ with l rows, corresponding to a text containing l tokens. Next, an MLP model projects $\mathbf{X}$ into a latent embedding matrix $\mathbf{Z}$ also with l rows. This $\mathbf{Z}$ matrix is used to compute $\boldsymbol{\beta}^{post,t} \in \mathbb{R}^{l \times c}$ through a normalizing flow (NF) operation. Each row of $\boldsymbol{\beta}^{post,t}$ represents the evidence count from the token's self-view, directly influencing the uncertainty of each token's prediction. In contrast to previous research, our approach includes the transmission of uncertainty from all tokens within the text to obtain the transmitted uncertainty $\boldsymbol{\beta}^{trans,t}$. Finally, we combine the sum of $\boldsymbol{\beta}^{post,t}$ and $\boldsymbol{\beta}^{trans,t}$ to generate the semantic matrix $\bar{\mathbf{p}}^{\mathbf{agg}} \in \mathbb{R}^{l \times c}$, representing the semantics of the l tokens. (b) Revised self-attention mechanism.

We choose PN as it does not require OOD data during training. In contrast, Prior Network [105], another representative EDL method, necessitates OOD data in training. Furthermore, even if OOD data is available, it may not cover all possible OOD scenarios. Thus, we opt for uncertainty transmission based on PN.

4.5.1 Our Token-Level Posterior Network

The PN, originally for image classification, is applied to NER for the first time to our knowledge. To better apply PN in NER, we first analyze the difference between tokens and samples (e.g., images). Concretely, tokens can be selected from specific sets, allowing for the calculation of token-level categorical distributions. In contrast, for samples, the vast and continuous potential space of unique samples makes it impractical to compute categorical distributions for every possible sample.

To apply PN into UE-NER and consider the above difference, we propose token-level PN, where we propose to calculate a unique categorical distribution for each token, rather than computing a single shared categorical distribution for all samples. This is because each token exhibits distinct semantic characteristics (e.g., "Paris" is more likely to represent a location than "August"), and thus needs individual categorical distributions. Concretely, a categorical distribution $\text{Cat}(\mathbf{p}_i)$ of i-th token in a text is the total occurrence of i-th token

in each of c classes given a training set. For example, the token "Apple" in the training data has 200 and 800 occurrences for the organization and food classes respectively, then "Apple" has categorical distribution as $[0, ..., 0.2, .., 0.8, 0...] \in \mathbb{R}^c$.

Then, since the classification is usually taken as a multinomial distribution, we can represent the classification as a posterior distribution as below,

$$\mathbb{P}(\mathbf{p}_i|y_i) \propto \mathbb{P}(y_i|\mathbf{p}_i) \times \mathbb{P}(\mathbf{p}_i) \tag{4.5}$$

we represent its prior distribution by a Dirichlet distribution $\mathbb{P}(\mathbf{p}_i) = \text{Dir}(\boldsymbol{\beta}^{prior})$, where $\boldsymbol{\beta}^{prior}$ is the parameter of the prior Dirichlet distribution. In practice, we set $\boldsymbol{\beta}^{prior} = \mathbf{1}$ for a flat equiprobable prior when the model brings no initial evidence. Due to the conjugate prior property, the posterior distribution can also be represented by a Dirichlet distribution: $\mathbb{P}(\mathbf{p}_i|y_i) = \text{Dir}(\boldsymbol{\beta}^{prior} + \boldsymbol{\beta}_i^{post})$. The $\boldsymbol{\beta}_i^{post}$ is taken as the evidence count for i-th token. To learn $\boldsymbol{\beta}_i^{post}$, PN firstly projects i-th token embedding $\mathbf{x}_i$ to a low-dimensional latent vector $\mathbf{z}_i = f(\mathbf{x}_i)$. Then, PN learns a normalized probability density $\mathbb{P}(\mathbf{z}_i|k;\theta)$ per class on this latent space. PN then counts the evidence for k-th class at $\mathbf{z}_i$ as below:

$$\boldsymbol{\beta}_{i,(k)}^{post} = N \times \mathbb{P}(\mathbf{z}_i|k;\theta) \times \mathbb{P}(k_i) \tag{4.6}$$

where $\mathbb{P}(k_i)$ is the probablity that i-th token belongs to k-th class, extracted from $\text{Cat}(\mathbf{p}_i)$. And $\boldsymbol{\beta}_i^{post} \in \mathbb{R}^c = [\beta_{i,(1)}^{post}, \beta_{i,(2)}^{post}, ..., \beta_{i,(c)}^{post}]$. The $\boldsymbol{\beta}_i^{post}$ can be understood as the evidence distribution for i-th tokens. For a text with l tokens, we can concatenate all l tokens' evidence distribution vector $\boldsymbol{\beta}^{post}$ and have $\boldsymbol{\beta}^{post,t} \in \mathbb{R}^{l \times c}$.

Difference to original posterior network. Compared to the original sample-level posterior network [10], which operates at the sample level, our token-level PN differs in two key ways: (1) We use a token-level categorical distribution instead of a sample-level categorical distribution shared among all samples. (2) We concatenate the $\boldsymbol{\beta}^{post}$ values for each of the l tokens to create a new matrix $\boldsymbol{\beta}^{post,t} \in \mathbb{R}^{l \times c}$ to facilitate uncertainty transmission in Sec. 4.5.2, a step not required in the original PN.

4.5.2 Our SLPN

Though the token-level PN counts the evidence given a token, it ignores the relation between tokens. Shown as Fig. 4.1, imagine that Token A comes from Text A, and Token B comes from Text B. Token A and Token B have the same predicted uncertainty in terms of token label when only considering the token itself. If the other tokens in Text A have more uncertainty than other tokens in Text B, then in this case, Token A should be more uncertain than Token B due to the impact of other tokens. Thus, we propose a Sequential Labeling Posterior Network (SLPN), which takes the uncertainty impact transmitted from other tokens into consideration.

Concretely, shown as Figure 4.2(a), a token embedding has accumulated all other token embeddings by the Bidirectional RNN [65] or transformer [155]. As a result, token uncertainty

should comprise two components: uncertainty originating from the token itself and uncertainty transmitted from other tokens. Since the uncertainty in EDL depends on the evidence count vector $\boldsymbol{\beta} \in \mathbb{R}^c$, we can represent the aggregated uncertainty $\boldsymbol{\beta}_i^{agg} \in \mathbb{R}^c$ for i-th token as below,

$$\boldsymbol{\beta}_i{}^{agg} = \boldsymbol{\beta}_i^{post} + \boldsymbol{\beta}_i^{trans} \tag{4.7}$$

where $\boldsymbol{\beta}_i^{post}$ is the uncertainty coming from the token itself and $\boldsymbol{\beta}_i^{trans} \in \mathbb{R}^c$ is the transmitted uncertainty from all tokens to i-th token in the text. The calculation of $\boldsymbol{\beta}_i^{post}$ is described in Sec. 4.5.1.

Calculation of impact transmission weight $\boldsymbol{\beta}_i^{trans}$. Since $\boldsymbol{\beta}_i^{trans}$ accumulates all the impact from all tokens in a text, we calculate $\boldsymbol{\beta}_i^{trans}$ in a way motivated by self-attention [155]. Concretely, we have three projector matrices $W_Q \in \mathbb{R}^{c\times p}$, $W_K \in \mathbb{R}^{c\times p}$ and $W_V \in \mathbb{R}^{c\times c}$ to get the query $Q \in \mathbb{R}^{l\times p}$, key $K \in \mathbb{R}^{l\times p}$ and value $V \in \mathbb{R}^{l\times c}$ as below,

$$\begin{aligned} &Q = \boldsymbol{\beta}^{post,t} W_Q, K = \boldsymbol{\beta}^{post,t} W_K \\ &V = softplus(\boldsymbol{\beta}^{post,t} W_V) \end{aligned} \tag{4.8}$$

where p is a pre-set dimension. Different from self-attention, we keep the shape of the V the same as $\boldsymbol{\beta}^{post,t}$, because the $\boldsymbol{\beta}^{post,t}$ has the evidence distribution and we want to avoid multiple projections that might lose the evidence distribution. Besides, we apply the $softplus$ activation function [146] to make sure the value of V is always greater than 0. We require evidence greater than 0 because EDL is an evidence acquisition process where each training sample adds support to learn higher order evidence distribution, and thus evidence can only be increased and not decreased [4, 159]. Then, we get the transmitted uncertainty $\boldsymbol{\beta}^{trans,t} \in \mathbb{R}^{l\times c}$ as below,

$$\boldsymbol{\beta}^{trans,t} = softmax(\frac{QK^T}{\gamma})V \tag{4.9}$$

where γ is the hyperparameter to rescale the weight to avoid gradient explosion. More explanation is given in Sec. 4.5.3.

Training Loss. Once we have obtained $\boldsymbol{\beta}^{agg}$ using Eq. 4.7, we train our SLPN model via below loss.

$$\begin{aligned} L = &\frac{1}{N}\sum_{i=1}^{N} \mathbb{E}_{\mathbf{p}_i^{agg}\sim \mathrm{Dir}(\mathbf{p}_i^{agg}|\boldsymbol{\alpha}_i^{agg})}[\mathrm{CE}(\mathbf{p}_i^{agg}, \mathbf{y}_i)] \\ &- \lambda \frac{1}{N}\sum_{i=1}^{N} \mathbb{H}(\mathrm{Dir}(\mathbf{p}_i^{agg}|\boldsymbol{\alpha}_i^{agg})) \end{aligned} \tag{4.10}$$

where $\boldsymbol{\alpha}_i^{agg} = \boldsymbol{\beta}_i^{agg} + \boldsymbol{\beta}^{prior}$ and the expected aggregated class probability of the i-th token calculated based on β_j^{agg} is below,

$$\bar{\mathbf{p}}_i^{agg} = \frac{\boldsymbol{\beta}_i^{agg} + \boldsymbol{\beta}^{prior}}{\sum_{k=1}^{c}(\beta_{i,(k)}^{agg} + \beta_k^{prior})} \tag{4.11}$$

Table 4.1: The table lists the applied entities for OOD and WS tasks. Recall that original ground-truth entities are $e^{og} = e^s + \hat{e}^g$ (used for OOD detection subtask), new ground-truth entities are $e^{ng} = e^s + \hat{e}^g + \hat{e}^p$ (used for WS detection subtask). The values in the brackets are the possible ground truth label values.

	e^s	$\hat{e}^g$	$\hat{e}^p$
Ground-truth entity labels	ID or OOD	ID or OOD	WS
OOD detection subtask usage	use (0 or 1)	use (0 or 1)	do not use (N/A)
WS detection subtask usage	use (0)	use(0)	use (1)

where $\boldsymbol{\beta}^{prior} \in \mathbb{R}^c$ is the vector with all default values as 1. As a result, the first item in Eq. 4.10 is the UCE loss in the token level like Eq. 4.3, and the second item in Eq. 4.10 is a regularization encouraging a smooth Dirichlet distribution for each token.

4.5.3 Explanation of Softplus

Since evidential learning is an evidence acquisition process, which means that every token in a training text contributes to learning an evidence matrix ($\boldsymbol{\beta}^{trans,t}$) [4, 139, 159], we expect that $\boldsymbol{\beta}^{trans,t}$ has all elements (e.g., all tokens' evidence in the respective class) greater than 0. Therefore, we expect every element of the evidential matrix ($\boldsymbol{\beta}^{trans,t}$) to be greater than 0.

Based on Eq 4.9, we understand that $\boldsymbol{\beta}^{trans,t}$ consists of two parts: the softmax part and V. If we expect $\boldsymbol{\beta}^{trans,t}$ to be greater than 0, the only potential negative case might be from V. Consequently, we anticipate that V is greater than 0. Therefore, we choose the Softplus function, which is defined as follows:

$$Softplus(x) = \log(1 + e^x). \tag{4.12}$$

Considering the formula of Softplus, it is always greater than 0. In addition to ensuring V is greater than 0 in Eq 4.8, we opt for Softplus as it helps prevent gradient explosion and gradient vanishing issues due to its smooth transition between the positive and negative parts of the input.

Table 4.2: Uncertainty estimation results MS_{ood+ws} on both OOD & WS tasks, the formula of MS_{ood+ws} is described in Eq. 4.13. The bold font annotates the best performance among a subregion. This bold font aligns with methodologies employed in similar studies on uncertainty estimation, including those detailed in Table 14 of [145] and Table 2 of [195].

Data	Model	weighted AUROC on both OOD & WS task					weighted AUPR on both OOD & WS task					F1
		Va.	Dis.	Al.	Ep.	En.	Va.	Dis.	Al.	Ep.	En.	
Mov-Sim	Dropout	-	-	68.66	71.09	72.11	-	-	27.98	34.06	31.28	**83.94**
	PN	**79.34**	54.41	66.56	**79.34**	65.14	40.88	16.90	30.27	40.88	27.72	82.43
	E-NER	77.58	59.05	76.82	77.58	77.61	36.40	20.44	35.72	36.40	36.05	70.63
	SLPN(w/o softplus)	60.12	39.66	45.35	60.12	37.33	28.72	16.57	23.47	28.72	19.36	66.95
	Ours(SLPN)	78.37	54.22	64.40	78.37	62.20	**47.23**	16.93	30.43	**47.23**	26.21	83.37
MIT-Res	Dropout	-	-	61.10	65.86	63.34	-	-	36.66	47.90	41.15	74.60
	PN	69.77	66.62	61.99	69.77	67.03	46.56	39.33	38.71	46.56	42.03	74.37
	E-NER	67.74	67.29	65.62	67.74	67.30	41.62	40.58	39.91	41.62	40.58	69.08
	SLPN(w/o softplus)	50.78	50.05	52.48	50.78	49.92	32.97	31.62	33.43	32.97	32.37	62.16
	Ours(SLPN)	**70.01**	49.14	57.17	**70.01**	53.02	**49.91**	32.08	35.23	**49.91**	34.85	**74.65**
Mov-Com	Dropout	-	-	55.82	56.44	56.13	-	-	17.01	18.68	17.40	**72.51**
	PN	72.65	68.07	71.08	72.65	69.43	28.88	22.93	27.47	28.88	25.99	70.13
	E-NER	77.93	73.77	77.68	77.93	75.55	34.32	25.34	29.48	34.32	27.99	67.21
	SLPN(w/o softplus)	60.77	54.44	57.91	60.77	55.56	25.18	20.93	24.32	25.18	22.71	66.05
	Ours(SLPN)	**81.31**	48.52	71.18	**81.31**	57.11	**38.47**	17.50	25.53	**38.47**	20.70	70.97

4.6 Experiments

4.6.1 Experimental Setup

Dataset Setup

Dataset. We apply three public datasets: (1) MIT-Restaurant (**MIT-Res**) dataset is in the restaurant domain with a total of 9181 texts with 8 semantic classes, excluding the "O" class. (2) Movie-Simple (**Mov-Sim**) dataset is in the movie domain with a total of 12,218 texts with 12 semantic classes, excluding the "O" class. (3) Movie-Complex (**Mov-Com**) dataset is also in the movie domain with a total of 9769 texts with 12 semantic classes, excluding the "O" class. These three datasets are provided in a common NER framework, Flair [1]. The criteria of the dataset choice are detailed in Sec. 4.6.3.

OOD entity construction & data split. Our OOD entities are constructed using the leave-out method. Specifically, given an NER dataset with different kinds of entity labels, we count the number of entities for each label. Subsequently, we select and leave out m labels with the lowest entity counts. This choice is made to ensure that there is a sufficient amount of data available for training and validation purposes. After applying the leave-out method, we represent the remaining labels as S^{in}, which includes c labels, and the corresponding text sets as D^{in}. Similarly, we represent the labels that were left out as S^{out}, which contains m OOD labels, and the corresponding text sets as D^{out}. All text samples in D^{in} are labeled only with entities from S^{in} and do not include any labels from S^{out}. Conversely, all text samples in D^{out} must contain at least one label from S^{out}.

Table 4.3: Size statistics on the three cases in three datasets.

Data	Model	e^{ng}	e^s	$\hat{e}^p$	$\hat{e}^g$
Mov-Sim	Dropout	4412	3055	488	869
	PN	4475	2974	551	950
	E-NER	4847	2665	923	1259
	SLPN (w/o softplus)	4991	2654	1067	1270
	Ours (SLPN)	4426	3060	502	864
MIT-Res	Dropout	7217	3793	1043	2381
	PN	7187	3667	1013	2507
	E-NER	7297	3456	1123	2718
	SLPN (w/o softplus)	7904	3646	1730	2528
	Ours (SLPN)	7237	3872	1063	2302
Mov-Com	Dropout	5551	3039	1019	1493
	PN	5772	3004	1240	1528
	E-NER	5689	2722	1157	1810
	SLPN (w/o softplus)	6043	2985	1511	1547
	Ours (SLPN)	5746	3045	1214	1487

We use 80% of the samples from D^{in} for training and 10% for validation. Our testing set comprises the remaining 10% of the samples from D^{in} and all samples from D^{out}.

Evaluation on OOD Detection

Our uncertainty estimation is evaluated via OOD detection at the entity level (e.g., "New York" is an entity with the label "LOC"). The reason for using entity-level evaluation is detailed in Sec. 4.6.3.

Wrong-span (WS) entities. However, OOD detection evaluation in the NER task faces challenges related to wrong-span (WS) entities. Unlike traditional image or text sample-level classification, NER tasks require the prediction of entity spans first. An entity may span one or several tokens. There are the following three cases related to OOD detection: (1) the predicted OOD entity exactly matches a true OOD entity; (2) the predicted OOD entity partially matches a true OOD entity on some tokens; (3) the predicted OOD entity does not match a true OOD entity on any tokens. We denote the second and third cases as "WS".

Three kinds of entities. Then, because these WS entities do not have ground truth ID/OOD labels, these WS entities are inapplicable for OOD detection evaluation. Besides, we are also interested in whether our UE-NER model can handle WS entity prediction as well. As a result, we aim to evaluate our UE-NER model Φ by both OOD detection and WS entity predictions. Because the entities applicable for evaluating WS entity prediction might be inapplicable for evaluating OOD detection, we divide the ground truth entities and predicted entities into three parts: (1) Unique predicted entities $\hat{e}^p$, which do not exist in the ground truth and thus are the WS entities; (2) Unique ground-truth entities $\hat{e}^g$, which are the entities that do not appear in the predicted entities; (3) Shared entities e^s, which are the predicted entities matching the ground truth.

Then, all predicted entities, including shared entities, are represented as $e^p = e^s + \hat{e}^p$. Original

ground-truth entities (without "WS" labels) are denoted as $e^{og} = e^s + \hat{e}^g$, and new ground-truth entities (including "WS" labels) are represented as $e^{ng} = e^s + \hat{e}^g + \hat{e}^p$.

Entities applied to OOD or WS detection. For NER OOD detection, the ground-truth labels in OOD detection should be binary, "ID" and "OOD" labels, while NER ground-truth labels have three: "ID", "OOD" and "WS" labels. As a result, we divide NER OOD detection into two subset for the evaluation. One subset has entities ($e^{og} = e^s + \hat{e}^g$) with "ID" and "OOD" for evaluating NER OOD detection, the other subset has $e^{ng} = e^s + \hat{e}^g + \hat{e}^p$ entities for evaluating WS detection. For OOD detection task, we take "OOD" labels as 1 and "ID" labels as 0. For WS detection, we take "WS" labels as 1, "ID" and "OOD" labels as 0. We list the applied entities of these two cases in Tab. 4.1.

Experimental Settings

Baselines. Because UE-NER is underexplored, we use three baselines in our experiments: (1) Dropout [30], which is an ensemble-based method to approximate BNN. It needs to run multiple times for the uncertainty estimation while our SLPN can get the estimated uncertainty by only running once. (2) PN [10], which has been revised into token-level PN for UE-NER task, introduced in Sec. 4.5.1. (3) E-NER [194] learns importance weights via evidence distribution and adds a regularization for increasing learned uncertainty of the wrong prediction.

Ablation Settings. Besides PN, we design SLPN (w/o softplus) for the ablation study. The SLPN (w/o softplus) removes the softplus in Eq. 4.8.

Uncertainty Metrics. We measure uncertainty estimation performance using five types of uncertainty. Specifically, Dissonance (Dis.) and vacuity (Va.) uncertainties are concepts proposed in the domain of evidential theory [139]. (1) Dissonance uncertainty refers to conflicting evidence, where the evidence for a particular class is similar to the evidence for other classes. (2) Vacuity uncertainty indicates a lack of evidence, where the evidence for all classes is of very small magnitude [87]. Besides, aleatoric (Al.) and epistemic uncertainty (Ep.) are proposed from the probabilistic view. (3) Aleatoric uncertainty arises from the inherent stochastic variability in the data generation process, such as noisy sensor data [22]. (4) Epistemic uncertainty stems from our limited knowledge about the data distribution, like OOD data. Moreover, we also consider (5) uncertainty calculated by entropy. We select the best-performing metric for each method from the five available uncertainty metrics. These five types of uncertainty are all measured via AUROC and AUPR [23, 61, 63, 105, 179, 195]. More details about the five uncertainty metrics are in Sec. 4.6.3.

For Tables 4.2, 4.4, and 4.5, we annotate the best performance within a subregion in bold font. This practice aligns with methodologies employed in similar studies on uncertainty estimation, including those detailed in Table 14 of [145] and Table 2 of [195].

Performance combined OOD and WS detection performance. Because we have

Table 4.4: Uncertainty estimation results on OOD task. The usage of bold font is the same as Table 4.2.

Data	Model	AUROC on OOD task					AUPR on OOD task					F1
		Va.	Dis.	Al.	Ep.	En.	Va.	Dis.	Al.	Ep.	En.	
Mov-Sim	Dropout	-	-	69.55	69.67	72.64	-	-	29.61	34.71	32.62	**83.94**
	PN	81.73	53.41	65.60	81.73	63.73	43.25	17.47	30.20	43.25	26.94	82.43
	E-NER	**84.20**	60.47	84.10	**84.20**	84.02	41.44	19.92	39.97	41.44	39.95	70.63
	SLPN(w/o softplus)	55.37	31.44	36.33	55.37	26.81	23.33	12.96	15.99	23.33	12.63	66.95
	Ours(SLPN)	81.29	53.59	64.10	81.29	61.27	**50.31**	17.56	30.77	**50.31**	25.57	83.37
MIT-Res	Dropout	-	-	58.01	64.26	61.08	-	-	39.97	54.36	45.52	74.60
	PN	73.50	69.44	60.98	73.50	70.03	53.39	45.16	43.07	53.39	47.88	74.37
	E-NER	**76.67**	75.76	74.53	**76.67**	75.76	51.27	49.79	49.11	51.27	49.79	69.08
	SLPN(w/o softplus)	44.30	43.92	46.22	44.30	41.69	32.66	33.74	33.78	32.66	31.41	62.16
	Ours(SLPN)	75.13	45.76	54.85	75.13	50.96	**58.93**	35.63	38.73	**58.93**	38.62	**74.65**
Mov-Com	Dropout	-	-	50.38	50.75	50.74	-	-	12.33	14.27	12.52	**72.51**
	PN	75.81	68.86	72.43	75.81	70.59	25.47	18.85	23.92	25.47	21.65	70.13
	E-NER	86.43	79.90	85.41	86.43	82.65	39.83	26.54	32.57	39.83	30.52	67.21
	SLPN(w/o softplus)	59.59	50.07	53.81	59.59	50.47	18.71	12.60	16.90	18.71	13.94	66.05
	Ours(SLPN)	**87.39**	44.28	71.63	**87.39**	55.35	**39.85**	12.47	21.41	**39.85**	15.24	70.97

Table 4.5: Uncertainty estimation results on WS task. The usage of bold font is the same as Table 4.2.

Data	Model	AUROC on WS task					AUPR on WS task					F1
		Va.	Dis.	Al.	Ep.	En.	Va.	Dis.	Al.	Ep.	En.	
Mov-Sim	Dropout	-	-	63.12	**79.96**	68.82	-	-	17.81	30.02	22.86	**83.94**
	PN	66.43	59.83	71.76	66.43	72.74	28.11	13.82	30.66	28.11	31.91	82.43
	E-NER	58.47	54.96	55.81	58.47	59.11	21.83	21.95	23.44	21.83	24.80	70.63
	SLPN(w/o softplus)	71.92	60.10	67.79	71.92	63.51	**42.11**	25.55	42.07	**42.11**	36.11	66.95
	Ours(SLPN)	60.60	58.09	66.26	60.60	67.87	28.43	13.07	28.34	28.43	30.11	83.37
MIT-Res	Dropout	-	-	**72.32**	71.70	71.54	-	-	24.61	24.39	25.28	74.60
	PN	56.25	56.40	65.63	56.25	56.18	21.84	18.24	22.93	21.84	20.87	74.37
	E-NER	40.25	41.21	38.21	40.25	41.27	11.94	12.22	11.61	11.94	12.24	69.08
	SLPN(w/o softplus)	64.43	62.97	65.66	64.43	67.27	33.62	27.16	32.68	33.62	**34.39**	62.16
	Ours(SLPN)	51.34	61.44	65.61	51.34	60.52	17.04	19.16	22.46	17.04	21.10	**74.65**
Mov-Com	Dropout	-	-	72.06	**73.41**	72.20	-	-	30.96	31.82	31.97	**72.51**
	PN	64.98	66.17	67.82	64.98	66.63	37.13	32.81	36.07	37.13	36.50	70.13
	E-NER	57.92	59.35	59.48	57.92	58.86	21.37	22.51	22.21	21.37	22.03	67.21
	SLPN(w/o softplus)	63.10	63.07	66.00	63.10	65.63	37.96	37.39	38.97	37.96	**40.03**	66.05
	Ours(SLPN)	66.05	59.16	70.04	66.05	61.52	35.00	30.12	35.88	35.00	34.41	70.97

OOD detection and WS detection tasks on NER uncertainty estimation, we propose to merge the results of the two tasks. This will enable us to determine which UE-NER model is better. As a result, we merge them by weighting the OOD detection results and WS detection results based on the size ratio between e^s and $\hat{e}^p$, as shown below.

$$MS_{ood+ws} = \frac{e^s}{e^s + \hat{e}^p} MS_{ood} + \frac{\hat{e}^p}{e^s + \hat{e}^p} MS_{ws} \tag{4.13}$$

Where MS_{ood+ws} represents the metric score weighted by the respective OOD task metric score MS_{ood} and the WS task metric score MS_{ws}.

4.6.2 Experimental Results

Our SLPN performs better than the baselines in weighted metric performance, which indicates that transmitted uncertainty from other tokens benefits the model performance. Table 4.2 shows that our SLPN outperforms the baselines in weighted metric performance, except for AUROC on Movie-Simple. Specifically, our SLPN surpasses the baselines in both AUROC and AUPR on the MIT-Restaurant dataset. For instance, our SLPN improves AUPR by 2.01 points compared to dropout and 3.25 points compared to PN. On the Movie-Simple dataset, the AUPR also indicates that our SLPN performs better than other methods, with an improvement of 6.35 points compared to PN. Although the AUROC on Movie-Simple does not exceed the baselines, the difference from PN is less than 1 point. Plus, on the Movie-Complex dataset, our work also surpasses the baselines, such as a 3.38 points improvement over the E-NER in AUROC. Taken together, these results demonstrate that the transmitted uncertainty from other tokens applied in SLPN benefits the model's performance.

The entity size distribution of our SLPN is similar to that of the baselines, except E-NER. Table 4.3 shows that the entity distributions for the three types of entities are similar among dropout, PN, and our SLPN. The relatively greater number of unique predicted entities $\hat{e}^p$ and the lower number of unique ground truth entities $\hat{e}^g$ compared to dropout suggests that our SLPN primarily improves OOD detection rather than WS detection. Consequently, future research can focus on enhancing WS detection or both of these detection tasks.

Additionally, we observe that E-NER has relatively fewer shared entities e^s. We speculate that this could be due to E-NER not demonstrating as powerful NER classification performance as dropout, PN, and our SLPN in these three datasets.

Our SLPN performs better than the baselines in OOD detection performance. Table 4.4 shows that E-NER performs better than our SLPN in Movie-Simple and MIT-Restaurant datsets, the E-NER sacrifices the NER classification performance. Among Dropout, PN and our SLPN, which have the similar high classification performance, our method performs better in OOD detection performance. For example, on the MIT-Restaurant dataset, our SLPN improves AUROC by 1.63 points compared to PN and 10.87 points compared to Dropout. However, on the Movie-Simple dataset, our SLPN has a difference of less than 1 point compared to PN, but our AUPR surpasses PN by 7.06 points.

Our SLPN performs unsatisfactorily compared to the baselines in WS detection performance. Although our SLPN performs very well in OOD detection, its performance in WS detection in Table 4.5 is unsatisfactory. However, the sizes of WS entities ($\hat{e}^p$) are very similar among dropout, PN, and our SLPN on both datasets. For example, the sizes of $\hat{e}^p$ are 1043, 1013, and 1063 for dropout, PN, and our SLPN, respectively. This means our SLPN performs unsatisfactorily in WS detection.

Our SLPN performs close or even better than the dropout in terms of the NER

task performance. From Table 4.5, our NER performance closely matches dropout, differing by less than 1 point in F1 scores on the Movie-Simple dataset. Notably, dropout is an ensemble-based approach known for enhancing model performance. Despite this, our SLPN achieves comparable or superior NER F1 scores, demonstrating its ability to enhance UE-NER performance while preserving the original NER model's effectiveness.

The activation function softplus is important to make the model performs in a stable way. When we remove the softplus operation (SLPN w/o softplus) and compare it with SLPN, we observe a significant performance decrease in both UE-NER and NER tasks. Table 4.2 indicates that NER F1 scores drop by over 10 points in both datasets, while UE-NER AUROC and AUPR scores decrease by more than 15 points. Thus, it is crucial to design the softplus operation in Eq. 4.8 to ensure β_i^{trans} remains positive.

4.6.3 More about Experiments

Criteria of Dataset Choice

We select the dataset based on two criteria: firstly, the dataset should contribute to reproducibility, and secondly, the dataset should not have an F1 score higher than 90%. We prioritize high reproducibility because we aim for our work to be replicable by others. We do not anticipate achieving an F1 score higher than 90%, as this would suggest that the dataset has already been thoroughly studied or that the model's uncertainty for that dataset is relatively low.

To meet the reproducibility criterion, we utilize the dataset provided by the Flair framework [1]. In adherence to the second criterion, we exclude CONLL_03 dataset from consideration due to its 94% F1 score in NER task. From the datasets listed in Flair framework [1], we randomly select two domains: the restaurant domain and the movie domain. For the restaurant domain, we opt for the MIT-Restaurant dataset. In the movie domain, Flair offers both a simple movie dataset and a complex movie dataset. We are interested in investigating whether there exists a tradeoff between uncertainty scores and F1 scores in UE-NER. Consequently, we select the simple-movie dataset and the complex-movie dataset, which exhibit higher and lower NER performance, as measured by the F1 score, in UE-NER, respectively. As for the tradeoff, after excluding the impact of different domains, we do not observe a significant tradeoff between the quality of uncertainty estimation (measured by AUROC) and NER task performance (measured by F1 score) when comparing the same method's AUROC and F1 between Mov-Sim and Mov-Com.

Reason of Entity-Level Evaluation

We choose entity-level evaluation instead of token level because it has more practical applications and is more commonly used in other NER works than token-level evaluation (e.g.,

"New" is a token with a label "b-LOC," and "York" is a token with a label "e-LOC"). Classifying "New" correctly and "York" incorrectly cannot lead to our desired correct entity.

Metrics

Below, we introduce the formulas used for the five metrics. Given a prediction from an EDL model, i.e., $\boldsymbol{\alpha}$, we have the total evidence $\alpha_0 = \sum_{k=1}^{c} \alpha_k$ (as in Eq.4.2) where c is the number of classes. The expected class probability is $\bar{\mathbf{p}} = \frac{\boldsymbol{\alpha}}{\alpha_0}$.

From the evidential view, we have dissonance and vacuity uncertainty for EDL-based models. The dissonance uncertainty in EDL is calculated via Eq. 5 in [195].

$$u^{\text{diss}} = \sum_{k=1}^{c} \frac{b_k \sum_{j \neq k} b_j \text{Bal}(b_j, b_k)}{\sum_{j \neq k} b_j} \tag{4.14}$$

with $b_k = \frac{\alpha_k - 1}{\alpha_0}$ and $\text{Bal}(b_j, b_k) = 1 - \frac{|b_j - b_k|}{b_j + b_k}$. It measures the uncertainty due to the conflicting evidence. The vacuity uncertainty in EDL is related to α_0 in Eq.4.2, which represents the total evidence,

$$u^{\text{vac}} = \frac{c}{\alpha_0} \tag{4.15}$$

From a probabilistic view, we have aleatoric uncertainty and epistemic uncertainty. The aleatoric uncertainty is calculated based on the projected or expected class probabilities,

$$u^{\text{alea}} = \frac{1}{\max_k \bar{p}_k} \tag{4.16}$$

The epistemic uncertainty is calculated based on total evidence in EDL-based models,

$$u^{\text{epis}} = \frac{1}{\alpha_0} \tag{4.17}$$

Because our vacuity uncertainty and epistemic uncertainty calculation are based on α_0 and are similar, they have the same sample rank regarding uncertainty score.

For dropout models, where the aleatoric and epistemic uncertainty are calculated from a probabilistic view, please refer to [52, 116].

We also report the entropy as the uncertainty score, which is calculated with the expected categorical distribution.

$$u^{\text{entropy}} = \mathbb{H}(\bar{\mathbf{p}}) \tag{4.18}$$

4.7 Conclusion

Incorrect NER predictions incur significant penalties. We primarily focus on UE-NER, which differs from prior uncertainty estimation methods that focus on sample-level labeling.

UE-NER centers on token-level sequential labeling, addressing the overlooked transmitted uncertainty from contextual tokens. We introduce SLPN to calculate uncertainty from both the token itself and contextual tokens, enhancing OOD detection in NER. Additionally, for OOD detection in NER, WS entities are not applicable. Thus, we divide the entities into two distinct subsets—one for OOD detection and the other for WS detection. Our experiments validate SLPN's effectiveness and the importance of considering uncertainty propagation in UE-NER.

Chapter 5

Can We Trust the Performance Evaluation of Uncertainty Estimation Methods in Text Summarization?[1]

Text summarization is a representative natural language generation (NLG) task. Due to the high penalty in wrongly predicted summaries in risk-critical domains, such as biased diagnosis based on incorrect symptom summaries in healthcare, uncertainty estimation on text summarization (UE-TS) has attracted researchers' attention. However, there remains a concern regarding UE-TS: can we trust the performance evaluation of UE-TS methods? This concern arises due to the dependency of uncertainty model metrics on NLG metrics and the diversity of NLG metrics. To answer this question and its associated questions, we introduce a UE-TS benchmark that incorporates thirty-one NLG metrics spanning four dimensions. The benchmark assesses the performance of two large language models and one pre-trained language model on two datasets in terms of their uncertainty estimation capabilities. Additionally, within this benchmark, we evaluate fourteen common uncertainty estimation methods in terms of their effectiveness. Our findings emphasize the importance of considering multiple uncorrelated NLG metrics and uncertainty estimation methods to ensure a reliable and efficient evaluation of uncertainty estimation methods.

5.1 Introduction

Text summarization [150] is a representative natural language generation (NLG) task that generates summaries for given texts. This study researches abstractive summarization [119], which is more flexible than extractive summarization [43]. In many practical scenarios (e.g.,

[1]J.He, M. Jin, L. Yu, C. Li, R. Yang, R. Jia, F. Chen, C. Lu. Can We Trust the Performance Evaluation of Uncertainty Estimation Methods in Text Summarization? Will be submitted to NIPS 2024.

finance and health), there are serious consequences if relying on incorrectly predicted summaries. For instance, an inaccurate financial report summary could lead to incorrect financial decisions, resulting in financial losses [37]. Consequently, the task of uncertainty estimation in text summarization (UE-TS) has garnered significant interest among researchers, seeking to measure the reflects the likelihood that a generated summary is low-quality [26, 36, 52]. However, an overlooked question persists regarding the reliability of the UE-TS evaluation framework. This concern stems from two reasons outlined below.

The first reason is that evaluation metrics for uncertainty estimation in NLG tasks rely on the alignment between the sample ranks obtained from the uncertainty scores and the sample rank based on the respective NLP metric scores. Concretely, commonly utilized metrics for uncertainty estimation in NLG tasks include forced-choice evaluation [52] and Prediction Rejection Ratio (PRR) [26, 108]. These metrics gauge uncertainty estimation by assessing the alignment between two types of sample ranks: those based on uncertainty scores from an uncertainty estimation method (e.g., entropy of generation semantics [78]) and those derived from a kind of NLP metric scores (e.g., ROUGE [94] in text summarization tasks). A higher alignment between two ranks indicates more accurate uncertainty scores and a more effective uncertainty estimation approach.

The second reason pertains to the label diversity in NLG tasks, such as text summarization, which leads to the utilization of various NLG metrics. Specifically, diverse generation tasks yield multiple acceptable responses [157]. For instance, in text summarization, summaries like "The cat chased the mouse" and "The mouse was pursued by the kitty" convey identical messages using different expressions. Due to this label diversity, it is unreasonable to expect NLG predictions to perfectly match the labels, unlike in NLU tasks [58, 63]. Consequently, NLG metrics vary, with each metric emphasizing different criteria related to the diverse labels. For example, metrics such as ROUGE [94] and BLEU [128] assess the similarity between the prediction and label based on n-gram overlap, while SummaC [80] examines the level of hallucination between the generation and reference text.

Collectively, previous evaluations of UE-TS models [26, 36, 52] have employed one or two uncertainty estimation metrics, yet each of these metrics depends solely on a single type of NLG metric. This limited evaluation raises a critical question:

> How does the choice of NLG metric affect the evaluation of uncertainty estimation methods in text summarization?

Answering this question is crucial for efficiently developing reliable UE-TS methods that can mitigate the risks associated with inaccurate summaries in practical applications, such as financial decision-making or healthcare. Moreover, we anticipate that our research could delve into other associated aspects, such as the relationship between uncertainty estimation methods and NLG metric scores, the impact of NLG metric diversity on future experimental design and analysis, and the effective selection of various types of uncertainty estimation

methods. Such exploration would enhance future experimental design and analysis methodologies. In essence, we aim to uncover the relationship between uncertainty estimation metrics and NLG metrics for evaluating the generated output in text summarization. To answer this and its associated questions, we introduce a UE-TS benchmark incorporating a diverse set of text generation metrics. The summary of our contribution is below.

- To the best of our knowledge, we are the first to raise a valid concern about the performance evaluation of UE-TS methods. To answer this question, we propose to evaluate UE-TS methods using different NLG metrics. These NLG metrics span four dimensions crucial for NLG evaluation, including coherence, consistency, fluency, and relevance, following [197].
- We present a UE-TS benchmark for evaluating uncertainty estimation through various NLG metric perspectives, marking the first endeavor to our knowledge. This benchmark assesses two Large Language Models (LLMs) and one Pre-trained Language Model (PLM) across two datasets focusing on UE-TS. Within this benchmark, we incorporate thirty-one NLG metrics and fourteen uncertainty estimation methods. To facilitate future research, **we will also provide intermediate results for each sample, alongside the code and data.** The intermediate results encompass the NLG metric scores and uncertainty scores for each sample.
- We have uncovered intriguing findings outlined in Sec. 5.6. Because text summarization is a representative NLG task, our findings in text summarization tasks could also motivate design and analysis on uncertainty estimation in other NLG tasks.

5.2 Related Work

Text summarization. There are two types of text summarization: extractive text summarization [101,170], which extracts the original sentences from the text for summarization, and abstractive text summarization [102,119], which directly generates summaries from the text. Due to the flexibility of abstractive text summarization, we focus on abstractive text summarization. Recent models for abstractive text summarization are typically divided into two categories: PLMs (e.g., BART [91]) and LLMs (e.g., Llama 2 [152]). To conduct comprehensive research on our question, we test abstractive summarization models from both PLM and LLMs.

Uncertainty estimation on text summarization. The UE-TS methods primarily fall into four categories [26]: information-based, density-based, ensemble-based, and prompt-based methods. Information-based methods use middle output (e.g., token probability) to obtain uncertainty scores. For example, [153] calculate uncertainty scores based on token logits in two ways: mean and Monte Carlo Dropout [161]. Similarly, [142,191] also calculate uncertainty based on token logits. Density-based methods leverage latent representations of

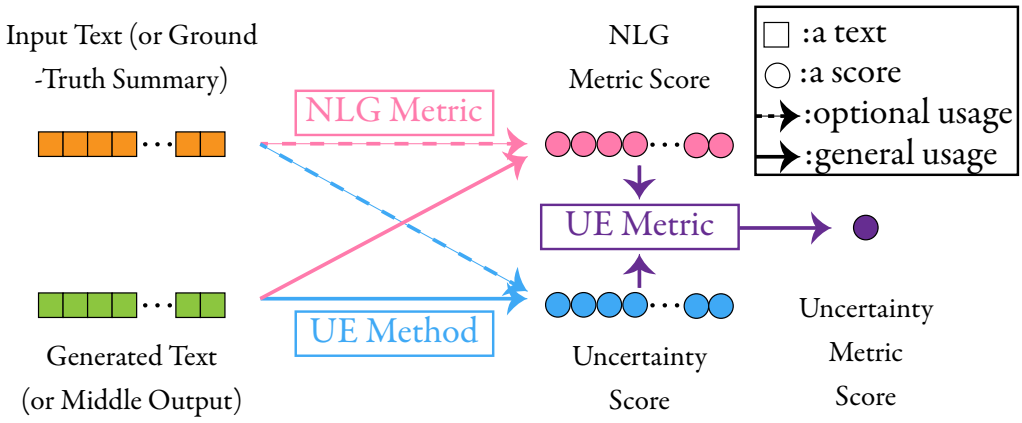

Figure 5.1: Diagram of the relationship between the Uncertainty Estimation (UE) metric, NLG metrics, and UE methods in the evaluation process. Specifically, the evaluation process for UE-TS methods involves using the generated texts (or intermediate outputs, such as token probabilities) and the optional input text (or ground-truth summary) to obtain NLG metric scores and uncertainty scores for all test samples, through an NLG metric and a UE method, respectively. Finally, the NLG metric scores and uncertainty scores for all testing samples are both inputted into a UE metric to obtain an uncertainty metric score of the UE method.

instances, which are further used to construct a probability density. For example, [134] detect low-quality generations by a density learned on the given sample embeddings. Ensemble-based methods use ensembles to approximate Bayesian Neural Networks (BNN) [116] or use variance of ensembled generation to obtain uncertainty scores. For example, dropout [30] is used in [36] to approximate BNN. Also, [18] obtain the uncertainty score by the variance of ensembling predictions. As for prompt-based methods, prompt-based methods refer to the methods that ask the generation model via a prompt to obtain the uncertainty score [72]. Some methods, like the SiCF score [52], integrate aspects of multiple uncertainty estimation methods, such as information and ensemble methods. To ensure comprehensive research on UE-TS methods, our benchmark includes representative methods from each category.

Performance evaluation of uncertainty estimation in text summarization. As for the uncertainty evaluation methods, [35, 36] obtain the uncertainty score by measuring the variance among ensembled generations based on an NLG metric, BLEU [128]. BLEU solely assesses uncertainty estimation method performance based on n-gram similarity. [77] compare the annotated expression of uncertainty between human annotations and annotations made by LLM based on an NLG metric, semantic similarity via SentenceBERT [133]. [89] evaluate UE-TS models based on the polarity score between prediction and ground truth summaries. [181, 196] evaluate uncertainty estimation methods solely using one NLG metric, ROUGE [94]. Only two NLG metrics, ROUGE and position accuracy, have been used separately for evaluation in [191]. Additionally, ROUGE and BERTScore [188], measuring semantic similarity, are both considered in [26, 52].

Thus, we observed that current evaluations of uncertainty estimation in text summarization rely on a limited set of NLG metric scores, which may lead to inconsistent performance rankings across other NLG metrics. Hence, our research aims to comprehensively explore the relationship between different uncertainty methods and various NLG metrics.

5.3 Metrics & Methods in Benchmark

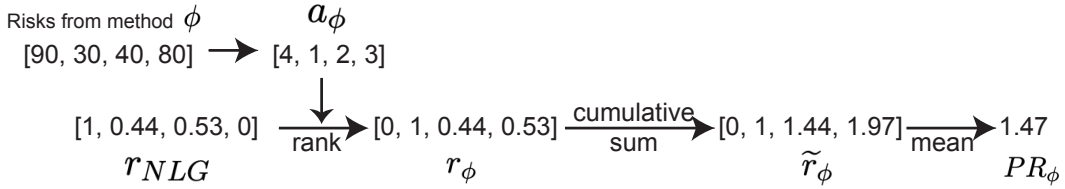

Figure 5.2: Diagram of the PR_ϕ calculation example with testing sample size $N = 4$. In this example, we have min-max normalized $\hat{s}_{NLG} = [0, 0.56, 0.47, 1]$, which is not drawn in the figure. Once we have obtained the sample rank a_ϕ based on a score list from method ϕ. We rerank r_{NLG} via a_ϕ to get r_ϕ. Then, we use Eq. 5.3 to cumulatively sum the elements and obtain $\widetilde{r}_\phi$. Finally, the PR_ϕ is the mean of $\widetilde{r}_\phi$.

5.3.1 Global View of Building Our Benchmark

To answer "How does the choice of NLG metric affect the evaluation of uncertainty estimation methods in text summarization?", our benchmark aims to explore the uncertainty estimation metric score across different NLG metrics for a given uncertainty estimation method. The relationship of these three items in the model evaluation process is shown in Figure 5.1. Below, we formalize the definition of the three scores we used in our work.

- **NLG metric score**: reflects the quality of a generation at the sample level via an NLG metric (e.g., ROUGE).
- **Uncertainty score**: reflects the likelihood that a model generation is low-quality at the sample level. The likelihood comes from an uncertainty estimation method (e.g., BNN).
- **Uncertainty estimation metric score**: reflects the performance of an uncertainty estimation method at the method level.

Regarding the NLG metrics, we evaluate fifteen different ones across four NLG evaluation dimensions, as proposed by [197]. For the uncertainty estimation methods to obtain uncer-

tainty scores, we examine ten common approaches outlined in [26]. For the uncertainty estimation metric, we adopt a widely used one, the Prediction Rejection Ratio (PRR) [26, 109]. Further details about these three components are provided below.

5.3.2 Uncertainty Estimation Metric: PRR

Although uncertainty estimation metrics, such as force-truth evaluation [52] and PRR [26, 109], rely on NLG metrics, we opt for PRR in our benchmark due to its efficiency. Unlike force-truth evaluation, which necessitates repeating NLG metric calculations ten times, PRR offers a more streamlined approach. It is formatted as follows:

$$PRR = \frac{PR_{uncertainty} - \frac{1}{\alpha}\sum_{i=1}^{\alpha} PR_{random}}{PR_{oracle} - \frac{1}{\alpha}\sum_{i=1}^{\alpha} PR_{random}} \tag{5.1}$$

where PR_ϕ is a scalar representing the cumulative risk, calculated based on a predicted sample rank a_ϕ from a method $\phi \in \{uncertainty, random, oracle\}$ and a list of NLG metric scores s_{NLG}. Specifically, the $s_{NLG} \in \mathbb{R}^N$ for N testing samples, is first min-max normalized to $\hat{s}_{NLG} \in \mathbb{R}^N$. Then, the risk for N testing samples is as follows,

$$r_{NLG} = 1 - \hat{s}_{NLG} \tag{5.2}$$

here, we assume that our chosen NLG metric score is positively correlated with generation performance, and common NLG metrics (e.g., ROUGE) meet this assumption. Thus, the NLG metric score (an element in $\hat{s}_{NLG}$)is negatively correlated with the risk (an element in r_{NLG}) of inaccurate generation. A higher risk indicates a higher chance of inaccurate generation. We then obtain r_ϕ by ranking r_{NLG} based on sample rank in a_ϕ. This process is shown in Fig. 5.2. Further, we obtain a cumulative risk vector $\widetilde{r}_\phi = [\widetilde{r}_{\phi,1}, \widetilde{r}_{\phi,2}, ..., \widetilde{r}_{\phi,N}]$. Its k-th element is calculated below,

$$\widetilde{r}_{\phi,k} = \sum_{j=1}^{k} r_{\phi,j} \tag{5.3}$$

finally, PR_ϕ is the mean of cumulative risk $\widetilde{r}_\phi$.

Because the risk is defined based on the normalized NLG metric score list $\hat{s}_{NLG}$ in Eq. 5.2, the $PR\phi$ is smaller if the predicted sample rank a_ϕ is more aligned with the sample rank based on the NLG metric score. A simple intuition is that if the predicted sample rank a_ϕ is identical to the sample rank based on r_{NLG}, PR_ϕ is minimized.

Then, we introduce PR_{oracle}, PR_{random}, and $PR_{uncertainty}$ as follows.

For PR_{oracle}, the predicted sample rank a_{oracle} is obtained via an oracle score vector $s_{oracle} = -1 \times s_{NLG}$. The PR_{oracle} is the lowest cumulative risk we can obtain because a_{oracle} is totally aligned with the sample rank obtained from the NLG metric score.

White-box	Information-based methods	Maximum Sequence Probability (**MSP**) Mean Token Entropy (**MTE**) Monte Carlo Sequence Entropy (**MCSE**)
	Density-based methods	Mahalanobis Distance (**MD**) Robust Density Estimation (**RDE**)
	Ensemble-based methods	Token-level Total Uncertainty (**T-TU**) Token-level Reverse Mutual Information (**T-RMI**) Sequence-level Total Uncertainty (**S-TU**) Sequence-level Reverse Mutual Information RMI (**S-RMI**)
	Prompt-based	**P(True)**
Black-box	Mixture of above four types	Number of Semantic Sets (**NumSets**) Eccentricity (**ECC**) Lexical Similarity (**LexSim**) Sum of Eigenvalues of the Graph Laplacian (**EigV**)

Table 5.1: A summary of the fourteen uncertainty methods that are used in our benchmark.

For PR_{random}, we generate a random permutation of N numbers each time. $\frac{1}{\alpha}\sum_{i=1}^{\alpha} PR_{random}$ represents the average of α different PR_{random} values, each calculated using a random permutation. In our experiments, we set α to 1000.

For $PR_{uncertainty}$, the predicted sample rank is obtained from the uncertainty score, which is calculated via an uncertainty estimation method. We list our uncertainty estimation methods in Sec. 5.3.3.

As a result, PRR in Eq. 5.1 calculates the relative risk between uncertainty scores from an uncertainty method and NLG scores from an NLG metric. The relative risk is normalized to random risk expectation $\frac{1}{\alpha}\sum_{i=1}^{\alpha} PR_{random}$. A higher PRR refers to a more accurate uncertain estimation model. This is because a smaller PR_{ϕ} leads to better alignment for method ϕ and the denominator in Eq. 5.1 is also negative. As a result, a smaller $PR_{uncertainty}$ leads to a larger PRR.

5.3.3 Uncertainty Estimation Methods

The uncertainty estimation methods utilized in our benchmark adhere to the framework established by [26], as the framework provides a publicly available uncertainty estimation tool, enhancing our results' reproducibility.

The uncertainty estimation methods can be divided into two kinds: white-box methods and black-box methods. White-box methods refer to uncertainty estimation methods using the intermediate output, model structure, or model parameters of the text summarization model (e.g., BART). Black-box methods refer to uncertainty estimation methods that require only the final text output from a text summarization model (e.g., GPT-3.5 [124]) for the uncertainty estimation. Below, we introduce our uncertainty estimation methods one by one, with a summary provided in Table 5.1.

White-Box Methods

The UE-TS methods primarily fall into four categories [26]: information-based, density-based, ensemble-based, and prompt-based methods.

Information-based methods utilize token probability to obtain uncertainty scores. Within this category, we employ the following methods:

(1) Maximum Sequence Probability (**MSP**): estimates uncertainty score as the log-probability of the generation in a greedy search way. It is calculated as the sum of log probabilities in each token.

(2) Mean Token Entropy (**MTE**): estimates the uncertainty score as the mean entropy for all tokens in a generation.

(3) Monte Carlo Sequence Entropy (**MCSE**): calculates the generation entropy estimations using Monte-Carlo estimation. It is the "predictive entropy" in [78].

Density-based methods utilize latent representations of instances to construct a probability density. Below, we list the related methods compared in our benchmark.

(4) Mahalanobis Distance (**MD**): calculates a Gaussian distribution for training samples. Then, the MD calculates the distance between the testing sample and the Gaussian distribution as the uncertainty score [84].

(5) Robust Density Estimation (**RDE**): improves over MD by reducing the dimensionality via principal component analysis decomposition [178].

Ensemble-based methods utilize ensembles to approximate Bayesian neural networks (BNN) or the variance of ensemble generations to obtain uncertainty scores. Below are the related methods we used:

(6) Token-level Total Uncertainty (**T-TU**): calculates the entropy of the predictive posterior at token level as the uncertainty score [52, 107].

(7) Token-level Reverse Mutual Information (**T-RMI**): uses reverse-KL divergence counterpart to the mutual information at token level as the uncertainty score [107].

(8) Sequence-level Total Uncertainty (**S-TU**): calculates the entropy of the predictive posterior at sequence level as the uncertainty score [52, 107].

(9) Sequence-level Reverse Mutual Information RMI (**S-RMI**): uses reverse-KL divergence counterpart to the mutual information at the sequence level as the uncertainty score [107].

Prompt-based methods refer to methods that prompt the generation model to obtain the uncertainty score [72]. We employ (10) **P(True)** [72], which takes the uncertainty score as the probability of asking whether the proposed answer is true or false. This method assumes that the generation model possesses superior zero-shot prompt abilities.

Relevance	ROUGE-L Spearman Kendall-Tau UniEval (Relevance) wo-GPT-3.5 (Relevance) wi-gt-GPT-3.5 (Relevance) wi-in-GPT-3.5 (Relevance) wi-ingt-GPT-3.5 (Relevance)
Consistency	BARTSCORE SummaC CTC UniEval (Consistency) wo-GPT-3.5 (Consistency) wi-gt-GPT-3.5 (Consistency) wi-in-GPT-3.5 (Consistency) wi-ingt-GPT-3.5 (Consistency)
Coherence	UniEval (Coherence) wo-GPT-3.5 (Coherence) wi-gt-GPT-3.5 (Coherence) wi-in-GPT-3.5 (Coherence) wi-ingt-GPT-3.5 (Coherence)
Fluency	UniEval (Fluency) wo-GPT-3.5 (Fluency) wi-gt-GPT-3.5 (Fluency) wi-in-GPT-3.5 (Fluency) wi-ingt-GPT-3.5 (Fluency)
Overall	UniEval (Overall) wo-GPT-3.5 (Overall) wi-gt-GPT-3.5 (Overall) wi-ingt-GPT-3.5 (Overall) wi-in-GPT-3.5(Overall)

Table 5.2: A summary of the twenty six NLG metrics that are used in our benchmark.

Black-Box Methods

For scenarios where only the final textual output is available, we utilize the following black-box uncertainty estimation methods, which have demonstrated effectiveness in previous studies [26].

(11) Number of Semantic Sets (**NumSets**): takes the number of diverse semantic interpretations for the generation as uncertainty score [95].

(12) Eccentricity (**ECC**): gets the uncertainty via calculating a distance between all eigenvectors that are informative embeddings of graph Laplacian [95].

(13) Lexical Similarity (**LexSim**): obtains the uncertainty scores via calculating mean similarity between all pairs of sampled generations [29].

(14) Sum of Eigenvalues of the Graph Laplacian (**EigV**): extends the NumSets from integer case into a continuous case, where the uncertainty score is calculated based on a matrix trace [95].

5.3.4 NLG Metrics

We chose thirty-one commonly used NLG metrics. To better understand the relationship between these NLG metrics, we categorize them into four dimensions [197]. We first introduce these four dimensions, then present all thirty-one NLG metrics with a summary in Table 5.2.

Dimensions for NLG Metrics

We choose four commonly used dimensions for NLG metrics, proposed in [197] and listed below.

Relevance refers to whether the generated text contains only the important information from the input text.

Consistency is the factual alignment between the generated text and the input text.

Coherence refers to whether all the sentences in the given generated text form a coherent body.

Fluency represents the quality of individual sentences in the generated text.

Additionally, we also consider the **overall** dimension that rates the generated texts based on all the above four dimensions.

Specific NLG Metrics

(1) **ROUGE-L**: measures the longest common subsequence between the generated text and ground-truth text [94]. It emphasizes relevance.

(2) **BARTSCORE**: uses an encoder-decoder pretrained model to compute a similarity score for each token in the generation with each token in the reference text [180]. It emphasizes consistency.

(3) **CTC**: uses a pretrained model to measure the alignment between the generation text and ground-truth text [20]. It emphasizes consistency.

(4) **SummaC**: uses pretrained models to segment both generated and input texts into sentence units and aggregate scores between pairs of sentences [80]. It emphasizes consistency.

(5) Spearman Correlation (**Spearman**) and (6) Kendall-Tau Correlation (**Kendall-Tau**) are designed to measure the semantic overlap between the model output and the reference text via text embeddings. They have relatively high correlations in the relevance dimension [99, 197].

(7-11) **UniEVAL**: takes NLG evaluation as a boolean question answering (QA) task and guides the model with different questions. UniEVAL [197] can use one evaluator to evaluate

one of the dimensions in Sec. 5.3.4.

(12-16) GPT-3.5 without dimension concepts **wo-GPT-3.5**: uses GPT-3.5 [124] to prompt about the generation quality. Among the prompts, we do not provide any prior knowledge about each dimension concept described in Sec. 5.3.4. This method compares the difference between the generated summaries and *ground-truth summaries*.

(17-21) GPT-3.5 with dimension concepts **wi-gt-GPT-3.5**: uses the GPT-3.5 [124] to prompt about the generation quality. However, the prior knowledge about each dimension concept is given in the prompt. This method compares the difference between the generated summaries and *ground-truth summaries*.

(22-26) **wi-in-GPT-3.5**: is very similar to wi-gt-GPT-3.5. The only difference is that this method compares the difference between the generated summaries and *input text*.

(27-31) **wi-ingt-GPT-3.5**: is very similar to wi-gt-GPT-3.5. The only difference is that this method compares generated summaries to the *input text and ground-truth summaries*.

5.4 Experiments

5.4.1 Experimental Settings

Dataset. We employ two widely-used text summarization datasets in our experiments. Firstly, we utilize the AESLC dataset [187], comprising 1,906 testing texts from the email domain. Secondly, we incorporate the XSUM dataset [120], featuring articles collected from the British Broadcasting Corporation (BBC) and encompassing 11,334 testing samples.

Related methods and metrics. We provide detailed descriptions of the uncertainty estimation methods in Sec.5.3.3. Additionally, an introduction to the NLG metrics is presented in Sec.5.3.4. By combining the uncertainty scores obtained from the uncertainty estimation methods with the NLG metric scores, we calculate the uncertainty metric score using the PRR method introduced in Sec. 5.3.2.

Our uncertainty measure in UE-TS is a combination of aleatoric and epistemic uncertainty. Regarding the sources of uncertainty in UE-TS, it originates from both aleatoric and epistemic uncertainty. Aleatoric uncertainty arises from inherent stochastic variability in the data generation process, such as incorrectly annotated ground truth summaries. On the other hand, epistemic uncertainty stems from our limited knowledge about the data distribution, such as when the testing text differs significantly from the training data distribution.

Implementation Details. To limit diverse generation, the temperature was set to 0, and a random seed of 42 was used. The experiments were conducted on a server equipped with a single A100 GPU. For the AESLC dataset, it took approximately 17 hours to run on GPT-3.5 generation and around 22 hours to run on Llama 2. For the XSUM dataset, the experiments

took approximately six times longer to run compared to those on AESLC. The total cost is about 1000 USD for the GPT-3.5-based generation and GPT-3.5-based evaluation. For the density-based method, we randomly sampled 1000 training samples with a fixed random seed from the respective dataset to represent the training data distribution, following the settings outlined in [26].

5.4.2 Experimental Results about NLG Metrics

Tables 5.7, 5.5, 5.3, 5.8, 5.6, and 5.4 present various uncertainty metric scores obtained from different uncertainty methods alongside different NLG metrics. Figures 5.6, 5.7, 5.8, 5.12, 5.13, and 5.14 illustrate the correlation of uncertainty methods in terms of uncertainty metric scores. Additionally, Figures 5.3, 5.4, 5.5, 5.9, 5.10, and 5.11 depict the correlation of NLG metrics with uncertainty metric scores.

Based on the information presented in these tables and figures, we can address our key question: "How does the choice of NLG metric affect the evaluation of uncertainty estimation methods in text summarization?" The answer is that **using different NLG metrics could lead to different ranks for uncertainty estimation methods**. Therefore, it is important to design uncertainty estimation metrics that are robust across various NLG metrics.

Next, we provide a detailed analysis from the perspectives of NLG metrics and uncertainty estimation methods, respectively.

Analysis Based on All NLG Metrics

Analysis from all five dimensions. Figures 5.3, 5.4, and 5.5 show the Spearman correlation between NLG metrics in a comprehensive view of the AESLC dataset. Figures 5.9, 5.10, and 5.11 show the Spearman correlation between NLG metrics in a comprehensive view of the XSUM dataset. From these figures, we can draw the following conclusions.

It is evident that evaluating uncertainty estimation models using different NLG metrics leads to variations in the performance ranking of these models. This discrepancy arises because there are no rows or columns in these figures where all elements are greater than 0.5. Thus, the reliability of evaluating uncertainty estimation models becomes a concern.

Additionally, some evaluations of uncertainty estimation models using different NLG metrics may result in different performance rankings. For instance, in Figure 5.10, it is observed that the correlation between ROUGE-L and wi-ingt-GPT-3.5 (overall) is -1, indicating a completely different ranking.

However, some evaluations of uncertainty estimation models using different NLG metrics could result in the same performance ranks. For instance, Figure 5.4 shows that the correlation between wi-ingt-GPT-3.5 (Fluency) and UniEval (Relevance) is 1, indicating identical

ranks.

Analysis on NLG Relevance Dimension

Figures 5.15, 5.16, 5.17, 5.27, 5.28, and 5.29 show the evaluation of UE-TS models from the relevance-dimension NLG metrics.

Based on the BART generation model, Figure 5.15 shows strongly positive correlations among ROUGE-L, Spearman, and Kendall-Tau. However, the Unieval (Relevance) and wo-GPT-3.5 (Relevance) show weak correlations with all other NLG metrics in the relevance dimension.

Also, based on the BART generation model, Figure 5.27 shows strongly positive correlations among ROUGE-L, Spearman, Kendall-Tau, and UniEval (Relevance). However, the wo-GPT-3.5 (Relevance) shows weak correlation with all other NLG metrics in the relevance dimension.

Based on the GPT-3.5 generation model, Figure 5.16 shows strongly positive correlations among Spearman, Kendall-Tau, UniEval (Relevance), and all GPT-3.5-based NLG metrics in the relevance dimension. However, Rouge-L shows weak correlation with all other NLG metrics in the relevance dimension. As for GPT-3.5-based NLG metrics, except for wi-in-GPT-3.5, we see that all other GPT-3.5-based models show identical performance ranks with each other. This implies that the relevance metric using GPT-3.5 is strongly related to the target text source.

Also, based on the GPT-3.5 generation model, Figure 5.28 shows a positive correlation among Spearman, Kendall-Tau, UniEval (Relevance), and all GPT-3.5-based NLG metrics in the relevance dimension, except for ROUGE-L.

Based on the Llama-2 generation model, in Figure 5.17, UniEval (Relevance) shows a strongly positive correlation with all other NLG metrics in the relevance dimension. This is because each element in the row of UniEval (Relevance) is greater than 0.5. Similarly, wo-GPT-3.5 (Relevance) also shows a strongly positive correlation with all other NLG metrics in the relevance dimension except for ROUGE-L. Additionally, we found that wi-in-GPT-3.5 shows weak correlation with other GPT-3.5-based NLG metrics. This weak correlation indicates that if we only provide the ground-truth summaries for the relevance evaluation, the relevance metrics may ignore many details.

Also, based on the Llama-2 generation model, Figure 5.29 shows that both UniEval (Relevance) and wo-GPT-3.5 (Relevance) show a positive correlation with other methods. The other metrics do not exhibit consistent positive or negative correlations with each other.

Besides, by comparing wo-GPT-3.5 (relevance) and wi-GPT-3.5 (relevance), where the only difference lies in whether GPT-3.5 is given the concept of relevance or not, their high positive correlation indicates that GPT-3.5 incorporates the concept of relevance. Consequently, the

difference in providing the relevance concept to GPT-3.5 is not apparent.

For the evaluation of UE-TS models using relevance-dimension NLG metrics, we have the following conclusions.

1. Generation models of the same type across different datasets could result in similar correlations among various methods.

2. Spearman and Kendall-Tau usually exhibit positive correlations. Therefore, in future experiments, choosing one of them is sufficient.

3. When employing LLMs as generation models, UniEval (Relevance) and wo-GPT-3.5 (Relevance) tend to exhibit positive correlations with most other NLG metrics. Therefore, one of them could serve as a representative NLG metric.

4. When utilizing LLMs as a type of relevance NLG metric, the choice of target text source can greatly impact the final conclusion. Specifically, using ground-truth summaries versus using input text as the target text source can result in different performance rankings.

5. When using LLMs as a type of relevance NLG metric, if both ground-truth summaries and input text are employed together as the target text source, the ground-truth summaries will dominate the metric results.

6. GPT-3.5 knows the concept of relevance. Consequently, the difference in providing the relevance concept to GPT-3.5 is not apparent.

Analysis on NLG Consistency Dimension

Figures 5.18, 5.19, 5.20, 5.30, 5.31, and 5.32 display the evaluation results of UE-TS models based on the consistency-dimension NLG metrics.

Specifically, based on BART generation, Figure 5.18 illustrates that UniEval (Consistency) and wo-GPT-3.5 (Consistency) exhibit a strongly positive correlation. SummaC and CTC show a positive correlation as well. Additionally, wo-GPT-3.5 (Consistency) displays a strongly positive correlation with BARTScore. However, there is no or even a negative correlation between the group of UniEval (Consistency) and wo-GPT-3.5 (Consistency) and the group of SummaC and CTC.

Also, based on BART generation, Figure 5.30 illustrates that UniEval (Consistency) and wo-GPT-3.5 (Consistency) exhibit a strongly positive correlation. SummaC and CTC show a positive correlation as well. However, SummaC displays a positive correlation with UniEval (Consistency) and wo-GPT-3.5 (Consistency), whereas CTC does not.

Based on GPT-3.5 generation, in Figure 5.19, positive correlations are found in a group consisting of CTC, UniEval (Consistency), and wi-in-GPT-3.5 (Consistency). Additionally, positive correlations are observed in another group comprising SummaC, wo-GPT-3.5 (Con-

sistency), wi-gt-GPT-3.5 (Consistency), and wi-ingt-GPT-3.5 (Consistency). However, these two groups display negative correlations. Thus, different metrics in the NLG consistency dimension can lead to different evaluation performance rankings of UE-TS models.

Also, based on GPT-3.5 generation, Figure 5.31 depicts positive correlations between CTC and UniEval (Consistency). Additional positive correlations are observed in a group comprising SummaC and all GPT-3.5-based NLG metrics. The only exception is that wi-ingt-GPT-3.5 (Consistency) and wo-ingt-GPT-3.5 (Consistency) exhibit negative correlations.

Based on Llama 2 generation, Figure 5.20 illustrates positive correlations in a group comprising BARTScore and all GPT-3.5-based NLG metrics. Additionally, positive correlations are found in a group consisting of CTC and UniEval (Consistency). However, negative correlations are observed for both of these groups. Regarding SummaC, it shows positive correlations with most of the first group except for wi-in-GPT-3.5 (Consistency) and negative correlations with the second group.

Also, based on Llama 2 generation, Figure 5.31 shows strong positive correlations in a group comprising CTC, UniEval (Consistency), and wo-GPT-3.5 (Consistency). Another set of strong positive correlations is found in a group including wi-gt-GPT-3.5, wi-in-GPT-3.5, and wi-ingt-GPT-3.5 (Consistency). However, these two groups exhibit negative correlations with each other. BARTSCORE and SummaC display negative correlations.

According to the above analysis, we can draw the following conclusions regarding the evaluation of UE-TS models using consistency-dimension NLG metrics.

1. Different metrics within the NLG consistency dimension can result in varying evaluation performance rankings of UE-TS models.

2. CTC tends to exhibit a positive correlation with SummaC or UniEval (Consistency). Thus, when faced with scenarios where we must choose between CTC, SummaC, or UniEval, we can opt for CTC due to its positive correlations with the other two in most cases.

3. For GPT-3.5-based NLG metrics, differences in the target text source could lead to discrepancies in some cases. Using ground-truth summaries as the target text source will have a more dominated impact compared to using input text as the target text source.

4. For GPT-3.5-based NLG metrics, GPT-3.5 might not fully comprehend the concept of consistency, as indicated by the negative correlation between wo-GPT-3.5 and wi-GPT-3.5 in Figure 5.34. However, the impact of understanding the concept of consistency is not as pronounced as the impact of using different target text sources.

5. For GPT-3.5-based NLG metrics, it is recommended to use wi-in-GPT-3.5 (Consistency) along with one of wo-gt-GPT-3.5, wi-gt-GPT-3.5 (Consistency), or wi-ingt-GPT-3.5.

Analysis on NLG Coherence Dimension

Figures 5.21, 5.22, 5.33 and 5.34 display the evaluation results of UE-TS models based on coherence-dimension NLG metrics. We do not illustrate correlations for BART generation models in the coherence dimension. This is because we only utilize UniEval (coherence) and wo-GPT-3.5 (coherence) for BART generations, and these two metrics may not adequately represent the correlation.

Based on GPT-3.5 generation, Figure 5.21 indicates that all GPT-3.5-based NLG metrics exhibit a strongly positive correlation with each other in terms of evaluating UE-TS models. However, UniEval (Coherence) demonstrates weak or even negative correlation with these GPT-3.5-based NLG metrics.

Also, based on GPT-3.5, Figure 5.33 demonstrates that a group comprising wo-GPT-3.5 (Coherence) and wi-ingt-GPT-3.5 (Coherence) exhibits a strongly positive correlation. Another strong positive correlation is observed in a group consisting of wi-gt-GPT-3.5 (Coherence) and wi-in-GPT-3.5 (Coherence). However, these two groups show no or even negative correlation with each other. Additionally, UniEval (Coherence) demonstrates weak or even negative correlation with these GPT-3.5-based NLG metrics.

Based on Llama 2 generation, Figure 5.22 illustrates that all GPT-3.5-based NLG metrics display positive correlations. Among them, wi-gt-GPT-3.5 shows a strongly positive correlation with other GPT-3.5-based NLG metrics except for wi-in-GPT-3.5 (Coherence). However, UniEval (Coherence) demonstrates weak or even negative correlation with these GPT-3.5-based NLG metrics.

Also, based on Llama 2 generation, Figure 5.34 indicates that all GPT-3.5-based NLG metrics exhibit positive correlations. wo-GPT-3.5 (Coherence) and wi-in-GPT-3.5 (Coherence) demonstrate a strongly positive correlation of 1. However, wi-gt-GPT-3.5 shows a relatively weak positive correlation with the other metrics. Regarding UniEval (Coherence), it displays a strongly positive correlation with wi-gt-GPT-3.5 (Coherence) but shows no or even negative correlation with the other three GPT-3.5-based NLG metrics.

According to the above analysis, we can obtain the following conclusions regarding the evaluation of UE-TS models using coherence-dimension NLG metrics:

1. It is difficult to determine which GPT-3.5-based metric is better in the coherence dimension. However, based on the strongly positive correlation, either wi-gt-GPT-3.5 or wi-in-GPT-3.5 could be a good choice.

2. UniEval (Coherence) exhibits weak or negative correlation with most of the GPT-3.5-based metrics. Therefore, UniEval (Coherence) could serve as a supplement to either wi-gt-GPT-3.5 or wi-in-GPT-3.5.

3. Based on Figures 5.33 and 5.34 on XSUM datasets, GPT-3.5 exhibits significant divergence between wo-GPT-3.5 and wi-gt-GPT-3.5. This suggests that GPT-3.5 itself might not fully

grasp the concept of coherence, and providing this concept could improve evaluation.

Analysis on NLG Fluency Dimension

Figures 5.23, 5.24, 5.35, and 5.36 display the evaluation results of UE-TS models based on fluency-dimension NLG metrics. We do not illustrate correlations for BART generation models in the fluency dimension. This is because we only utilize UniEval (fluency) and wo-GPT-3.5 (fluency) for BART generations, and these two metrics may not adequately represent correlation.

Using GPT-3.5 as a generation model, Figure 5.23 indicates that wi-in-GPT-3.5 (Fluency) has a negative correlation with wo-GPT-3.5, wi-gt-GPT-3.5, and wi-ingt-GPT-3.5 (Fluency). UniEval (Fluency) demonstrates weak or negative correlation with these GPT-3.5-based NLG metrics.

Also, using GPT-3.5 for generation, Figure 5.35 illustrates that all the GPT-3.5-based NLG metrics except wi-ingt-GPT-3.5 (Fluency) exhibit strongly positive correlations. In contrast, these three GPT-3.5-based NLG metrics show negative correlation to UniEval (Fluency) and wi-ingt-GPT-3.5 (Fluency).

Using Llama 2 as a generation model, Figure 5.24 demonstrates that all five metrics have positive correlations. Among them, wi-gt-GPT-3.5, wi-in-GPT-3.5, and wi-ingt-GPT-3.5 (Fluency) exhibit strongly positive correlations.

Figure 5.36 illustrates that UniEval (Fluency) has strongly positive correlations with wo-GPT-3.5 (Fluency) and wi-gt-GPT-3.5 (Fluency). However, it exhibits strongly negative correlation with wi-in-GPT-3.5 (Fluency) and wi-ingt-GPT-3.5 (Fluency). Among the GPT-3.5-based NLG metrics, only wi-gt-GPT-3.5 (Fluency) and wi-in-GPT-3.5 (Fluency) show positive correlations, while the other correlations are weak or negative.

Based on the above findings, we can conclude the following observations regarding the fluency dimension.

1. It is challenging to determine which NLG metrics consistently exhibit positive correlations with others. However, based on the positive correlations, we recommend wi-gt-GPT-3.5 (Fluency) and wi-ingt-GPT-3.5 (Fluency) due to their correlation patterns.

2. Because of the negative correlations between UniEval (Fluency) and GPT-3.5-based NLG metrics, we also recommend using UniEval (Fluency) as a supplement to a GPT-3.5-based NLG metric.

3. In the context of GPT-3.5-based NLG metrics, the choice of target text source influences the performance ranking of UE-TS models. Furthermore, employing ground-truth summaries as the target text source exerts a more significant impact compared to using input text.

4. The same generation methods cannot achieve similar correlations for fluency NLG di-

mension. Consequently, correlations within fluency dimensions are more closely tied to the dataset.

5. Due to the positive or negative correlations between wo-GPT-3.5 (Fluency) and wi-GPT-3.5 (Fluency), it is difficult to determine whether GPT-3.5 itself understands the concept of fluency.

Analysis on NLG Overall Dimension

Figures 5.25, 5.26, 5.37, and 5.38 display the evaluation results of UE-TS models based on overall-dimension NLG metrics. Correlations for BART generation models in the overall dimension are not illustrated. This is because we only utilize UniEval (overall) and wo-GPT-3.5 (overall) for BART generations, and these two metrics may not adequately represent correlation.

Utilizing GPT-3.5 as the generation model, Figure 5.25 shows that all GPT-3.5-based NLG metrics have identical performance rankings for UE-TS models. Additionally, these GPT-3.5-based NLG metrics exhibit a strongly positive correlation with UniEval (Overall).

Also, utilizing GPT-3.5 as the generation model, Figure 5.25 illustrates that wo-GPT-3.5 (Overall) has positive correlations with the other three GPT-3.5-based NLG metrics. Furthermore, strongly positive correlations are found between wo-GPT-3.5 (Overall) and wi-gt-GPT-3.5 (Overall). Another strongly positive correlation is observed between wi-in-GPT-3.5 (Overall) and wi-ingt-GPT-3.5 (Overall).

Utilizing Llama 2 as the generation model, Figure 5.26 indicates that all GPT-3.5-based NLG metrics have strongly positive correlations. However, UniEval (Overall) exhibits negative correlations with these GPT-3.5-based NLG metrics.

Also, utilizing Llama 2 as the generation model, Figure 5.38 shows that almost all five metrics have positive correlations, with the exceptions being that wi-in-GPT-3.5 (Overall) has a negative correlation with UniEval (Overall) and wo-GPT-3.5 (Overall).

Based on the above findings, we can conclude the following regarding the overall dimension.

1. When it comes to the GPT-3.5-based NLG metrics, given their positive correlations with other metrics, it is recommended to use either wi-in-GPT-3.5 (Overall) or wi-gt-GPT-3.5 (Overall).

2. The UniEval (Overall) exhibits high correlations with other GPT-3.5-based methods in most cases and negative correlations in a few instances. It could serve as an optional supplementary tool for GPT-3.5-based NLG metrics.

3. Due to the positive correlations between wo-GPT-3.5 and wi-GPT-3.5 NLG metrics in most cases, it suggests that GPT-3.5 itself grasps the concept of overall.

4. Unlike previous NLG metric dimensions, it is difficult to determine which one dominates the evaluated performance ranks between using ground-truth summaries and using input text.

5.4.3 Experimental Results about Uncertainty Estimation Methods

Analysis on ensemble-based uncertainty estimation methods. In the case of ensemble-based uncertainty estimation methods, T-TU, S-TU, and S-RMI generally exhibit positive correlations with all other white-box uncertainty estimation methods, except for T-RMI. This trend is evident in Figures 5.6 and 5.12. Additionally, T-RMI demonstrates negative correlations with T-TU and S-TU, as well as a weakly positive correlation with S-RMI. Consequently, we propose that for future applications of uncertainty estimation methods, focusing on one of T-TU, S-TU, and S-RMI could be advantageous, with T-RMI serving as a supplementary baseline.

Analysis on information-based uncertainty estimation methods. In the realm of information-based uncertainty estimation methods, MSP and MCSE typically exhibit strongly positive correlations with each other, as evidenced in Figures 5.6, 5.8, 5.12, and 5.14.

Conversely, MTE exhibits inconsistently strong positive correlations with MSP and MCSE. For instance, while Figure 5.8 demonstrates a weakly positive correlation between them, Figure 5.12 depicts a strongly positive correlation. Therefore, we recommend that in future comparison of uncertainty estimation methods, when considering MSP and MCSE, opting for one of them and optionally utilizing MTE as a supplementary baseline could be beneficial.

Analysis on density-based uncertainty estimation methods. In the domain of density-based uncertainty estimation methods, MD and RDE typically demonstrate strongly positive correlations with each other. This trend is observed in Figures 5.6, 5.12, and 5.14, with the exception of a weakly positive correlation in Figure 5.8.

Hence, we propose that in future applications of uncertainty estimation methods, utilizing either MD or RDE alone would suffice rather than using both of them as the baselines.

Analysis on prompt-based uncertainty estimation methods. In the case of prompt-based uncertainty estimation methods, P(True) typically exhibits a weak or negative correlation with other uncertainty estimation methods. This trend is evident from Figures 5.8 and 5.14.

Analysis on black-box uncertainty estimation methods. In the realm of black-box uncertainty estimation methods, ECC, LexSim, and EigV typically exhibit positive correlations with each other, with ECC and EigV showing particularly strong correlation. Conversely, NumSets demonstrates a weak or negative correlation with ECC, LexSim, and EigV. These trends are evident in Figures 5.7 and 5.13.

Hence, we recommend that in future applications of uncertainty estimation methods in-

volving ECC, LexSim, and EigV, opting for one of them and employing NumSets as a complementary measure could be beneficial.

Analysis on specific uncertainty estimation methods. Besides, we observed that each UE method consistently achieves positive UniEval (Consistency) scores across all eight methods listed in Table 5.7. This indicates that all of the aforementioned methods outperform random ranking in terms of the NLG metric, UniEval (Consistency).

Each uncertainty estimation method exhibits dissatisfaction performance with negative ROUGE-L scores across all eight methods in Table 5.8. This suggests that all the methods listed above perform worse than random ranking in terms of the NLG metric, ROUGE-L (Consistency).

Each uncertainty estimation method demonstrates strong performance, achieving positive scores for both UniEval (Consistency) and UniEval (Coherence) across all four methods presented in Table 5.5. This indicates that all aforementioned methods outperform random ranking in terms of both consistency and coherence according to UniEval assessment.

5.4.4 More about Experiments

In this subsection, we list more experimental results as below.

5.5 Conclusion

Due to the dependency of uncertainty model metrics on NLG metrics and the diversity of NLG metrics, we propose a UE-TS benchmark to answer the question, "How does the choice of NLG metric affect the evaluation of uncertainty estimation methods in text summarization?" The comprehensive benchmark is built on two datasets with fourteen uncertainty estimation methods and thirty-one NLG metrics. The uncertainty estimation methods cover both white-box methods, including four different types of uncertainty estimation methods, and black-box methods. The NLG metrics span four different NLG evaluation dimensions. Additionally, our benchmark includes the evaluation of one black-box LLM, one white-box LLM, and one PLM for the text summaries. While it is evident that the current UE-TS model evaluations lack reliability, our work contributes to a broader understanding of the relationship between NLG metrics and uncertainty estimation methods. This understanding can guide future research and inform the efficient development of more effective uncertainty estimation methods and evaluation protocols for UE-TS or other NLG tasks.

NLG Metrics	**MSP**	**MTE**	**MCSE**	**MD**	**RDE**	**P(True)**
ROUGE-L	0.2107	0.1668	0.2082	0.2650	0.1608	-0.0233
BARTSCORE	0.0372	0.2015	0.0418	0.2451	0.1573	0.0615
SummaC	-0.1301	-0.0440	-0.1189	0.0739	-0.0182	0.0998
CTC	0.0736	-0.1515	0.0685	0.0457	0.0347	-0.1074
Spearman	0.0656	0.1430	0.0640	0.1429	0.0929	-0.0080
Kendall-Tau	0.0649	0.1412	0.0630	0.1404	0.0904	-0.0084
UniEval (Relevance)	-0.0572	0.0769	-0.0309	0.1995	-0.0334	-0.2365
UniEval (Consistency)	0.1408	0.1007	0.1143	0.0224	0.1392	-0.3789
UniEval (Coherence)	0.1804	0.1802	0.1932	0.2163	0.2332	-0.7554
UniEval (Fluency)	-0.0524	-0.3809	0.0079	0.0568	-0.1954	-0.2343
UniEval (Overall)	0.0300	0.0000	0.0563	0.1702	0.0148	-0.4410
wo-GPT-3.5 (Relevance)	-0.0863	0.2315	-0.0451	0.1784	-0.0533	0.1029
wo-GPT-3.5 (Consistency)	-0.0280	0.1825	0.0025	0.2586	0.0157	0.0849
wo-GP-T3.5 (Coherence)	-0.0796	0.0771	-0.0479	0.1672	-0.0712	0.0969
wo-GPT-3.5 (Fluency)	-0.0148	0.1498	0.0023	0.1992	-0.0347	0.0615
wo-GPT-3.5 (Overall)	-0.0634	0.0976	-0.0794	0.1807	0.0187	0.0399
wi-gt-GPT-3.5 (Relevance)	-0.2205	0.0841	-0.2692	0.0835	0.0833	-0.1177
wi-gt-GPT-3.5 (Consistency)	-0.2533	-0.0078	-0.2726	-0.0076	-0.0268	-0.0773
wi-gt-GPT-3.5 (Coherence)	-0.2991	-0.0493	-0.2979	0.0333	-0.0597	-0.1096
wi-gt-GPT-3.5 (Fluency)	-0.2344	-0.0392	-0.2084	0.1485	0.0882	-0.3175
wi-gt-GPT-3.5 (Overall)	-0.1829	0.0045	-0.2133	-0.0836	-0.0842	-0.1202
wi-in-GPT-3.5 (Relevance)	0.0185	0.0798	0.0735	0.0608	-0.0820	-0.0239
wi-in-GPT-3.5 (Consistency)	0.1015	0.1054	0.1041	0.1972	0.0545	-0.1937
wi-in-GPT-3.5 (Coherence)	-0.0170	0.0653	-0.0206	0.0283	-0.1453	-0.2302
wi-in-GPT-3.5 (Fluency)	0.0808	-0.1020	0.0955	0.1685	0.0308	-0.4904
wi-in-GPT-3.5 (Overall)	-0.1816	0.1285	-0.1818	0.0457	-0.0564	-0.0854
wi-ingt-GPT-3.5 (Relevance)	-0.1114	0.1486	-0.1015	0.1182	0.0500	-0.0379
wi-ingt-GPT-3.5 (Consistency)	-0.1391	0.1558	-0.1462	0.0934	0.0114	-0.0660
wi-ingt-GPT-3.5 (Coherence)	-0.1079	0.0909	-0.0959	0.1531	0.0058	-0.1189
wi-ingt-GPT-3.5 (Fluency)	-0.0283	0.0799	0.0065	0.1402	-0.0077	-0.2216
wi-ingt-GPT-3.5 (Overall)	-0.1149	0.0485	-0.1011	-0.0094	-0.1268	-0.0594
Col Mean	-0.0451	0.0634	-0.0364	**0.1204**	0.0092	-0.1263

Table 5.3: Main results of the relationship between the uncertainty estimation methods and NLG metrics on AESLC dataset using generation from Llama 2.

NLG Metrics	**MSP**	**MTE**	**MCSE**	**MD**	**RDE**	**P(True)**
ROUGE-L	0.1300	0.1171	0.1257	0.0096	-0.0232	-0.1698
BARTSCORE	0.1657	0.2000	0.1789	0.2132	0.1891	-0.1828
SummaC	-0.0659	-0.0580	-0.0642	-0.1556	-0.2557	0.0392
CTC	0.0448	-0.0112	0.0699	-0.0021	-0.0759	-0.3161
Spearman	0.0189	0.0516	0.0094	0.0569	0.0190	-0.2855
Kendall-Tau	0.0138	0.0437	0.0034	0.0495	0.0151	-0.2735
UniEval (Relevance)	0.4496	0.4654	0.4586	0.3709	0.1655	-0.8063
UniEval (Consistency)	0.5711	0.5589	0.5746	0.4069	0.2830	-0.7399
UniEval (Coherence)	0.7198	0.7069	0.7219	0.5884	0.2795	-1.0973
UniEval (Fluency)	0.3714	0.3008	0.3969	0.3526	0.2316	-0.5880
UniEval (Overall)	0.6020	0.5904	0.6113	0.4893	0.2567	-0.9452
wo-GPT-3.5 (Relevance)	0.3447	0.3852	0.3756	0.2945	-0.0065	-0.3841
wo-GPT-3.5 (Consistency)	0.0789	0.1104	0.0990	0.0993	0.0142	-0.0140
wo-GP-T3.5 (Coherence)	-0.0197	-0.0238	-0.0012	-0.0316	0.0102	0.0612
wo-GPT-3.5 (Fluency)	0.1701	0.1997	0.1935	0.1850	-0.0118	-0.0961
wo-GPT-3.5 (Overall)	0.1249	0.1626	0.1442	0.1562	0.0138	-0.1296
wi-gt-GPT-3.5 (Relevance)	-0.1070	-0.1089	-0.0933	-0.1349	-0.1222	-0.1304
wi-gt-GPT-3.5 (Consistency)	-0.0985	-0.1148	-0.0832	-0.0625	-0.0538	-0.0531
wi-gt-GPT-3.5 (Coherence)	-0.0546	-0.0504	-0.0270	-0.0813	-0.0471	-0.0909
wi-gt-GPT-3.5 (Fluency)	-0.1158	-0.1192	-0.1139	-0.1233	-0.1168	-0.1173
wi-gt-GPT-3.5 (Overall)	-0.0327	-0.0285	-0.0130	-0.0718	-0.0911	-0.0383
wi-in-GPT-3.5 (Relevance)	-0.0105	-0.0084	-0.0035	-0.0576	-0.0280	-0.0266
wi-in-GPT-3.5 (Consistency)	-0.0276	-0.0289	-0.0069	-0.0458	-0.0211	0.0216
wi-in-GPT-3.5 (Coherence)	-0.0454	-0.0456	-0.0234	-0.0526	-0.0171	0.0429
wi-in-GPT-3.5 (Fluency)	-0.1349	-0.1423	-0.1161	-0.1589	-0.1039	0.0210
wi-in-GPT-3.5 (Overall)	-0.1067	-0.1069	-0.0861	-0.1170	-0.0915	-0.0361
wi-ingt-GPT-3.5 (Relevance)	-0.0269	-0.0038	-0.0295	-0.0894	-0.1017	-0.0178
wi-ingt-GPT-3.5 (Consistency)	-0.0940	-0.0760	-0.0917	-0.1211	-0.0636	-0.0263
wi-ingt-GPT-3.5 (Coherence)	-0.0587	-0.0155	-0.0530	-0.1187	-0.1089	-0.0441
wi-ingt-GPT-3.5 (Fluency)	-0.1545	-0.1254	-0.1790	-0.1492	-0.1588	-0.0200
wi-ingt-GPT-3.5 (Overall)	-0.0362	-0.0156	-0.0098	-0.0193	-0.0316	-0.0162
Col Mean	0.0844	0.0906	**0.0957**	0.0542	-0.0017	-0.2084

Table 5.4: Main results of the relationship between the uncertainty estimation methods and NLG metrics on XSUM dataset using generation from Llama 2.

NLG Metrics	**NumSets**	**ECC**	**LexSim**	**EigV**
ROUGE-L	0.0415	0.1071	0.2121	0.0964
SummaC	0.0919	-0.1259	-0.1362	-0.0994
CTC	0.0387	0.0999	0.0506	0.1132
Spearman	0.0326	0.0101	0.1344	0.0112
Kendall-Tau	0.0326	0.0102	0.1324	0.0114
UniEval (Relevance)	0.1131	-0.0728	-0.0165	-0.0747
UniEval (Consistency)	0.0559	0.0898	0.1260	0.1211
UniEval (Coherence)	0.0992	0.1668	0.2259	0.1754
UniEval (Fluency)	-0.0252	-0.1101	-0.2024	-0.0921
UniEval (Overall)	0.0946	-0.0379	-0.0055	-0.0284
wo-GPT-3.5 (Relevance)	0.0714	-0.0908	0.1165	-0.1115
wo-GPT-3.5 (Consistency)	0.0977	-0.0382	0.0806	-0.0488
wo-GP-T3.5 (Coherence)	0.1004	-0.0981	0.0624	-0.1043
wo-GPT-3.5 (Fluency)	0.1217	-0.1114	0.0466	-0.1265
wo-GPT-3.5 (Overall)	0.1008	-0.0548	0.0785	-0.0734
wi-gt-GPT-3.5 (Relevance)	0.1465	-0.0595	0.1097	-0.0893
wi-gt-GPT-3.5 (Consistency)	0.1002	-0.0101	0.0344	-0.0330
wi-gt-GPT-3.5 (Coherence)	0.1418	-0.0267	0.0512	-0.0519
wi-gt-GPT-3.5 (Fluency)	0.1501	0.0117	0.1928	0.0091
wi-gt-GPT-3.5 (Overall)	0.1226	-0.0892	0.0206	-0.0953
wi-in-GPT-3.5 (Relevance)	0.0303	-0.0291	0.0783	-0.0334
wi-in-GPT-3.5 (Consistency)	0.0792	0.1707	0.2616	0.1626
wi-in-GPT-3.5 (Coherence)	0.1428	0.0940	0.2092	0.0840
wi-in-GPT-3.5 (Fluency)	0.0167	0.1683	0.0820	0.1765
wi-in-GPT-3.5 (Overall)	0.0774	-0.1700	0.0117	-0.1711
wi-ingt-GPT-3.5 (Relevance)	0.1462	-0.1226	0.0659	-0.1473
wi-ingt-GPT-3.5 (Consistency)	0.0677	-0.1344	0.0135	-0.1543
wi-ingt-GPT-3.5 (Coherence)	0.0395	-0.1203	0.0317	-0.1438
wi-ingt-GPT-3.5 (Fluency)	0.0971	-0.0075	0.1304	-0.0011
wi-ingt-GPT-3.5 (Overall)	0.1281	-0.0642	0.0760	-0.0918
Col Mean	**0.0851**	-0.0215	0.0758	-0.0270

Table 5.5: Main results of the relationship between the uncertainty estimation methods and NLG metrics on AESLC dataset using generation from GPT-3.5.

NLG Metrics	**NumSets**	**ECC**	**LexSim**	**EigV**
ROUGE-L	-0.0207	0.0867	0.0759	0.0565
SummaC	0.0445	-0.0485	-0.1196	-0.0297
CTC	0.0058	0.1267	0.0278	0.0956
Spearman	-0.0045	-0.0721	-0.1082	-0.0781
Kendall-Tau	-0.0046	-0.0664	-0.1084	-0.0716
UniEval (Relevance)	0.0468	-0.0740	-0.1099	-0.0728
UniEval (Consistency)	-0.0325	0.3751	0.4828	0.3161
UniEval (Coherence)	-0.0033	0.3327	0.4128	0.2822
UniEval (Fluency)	0.0573	0.2495	0.1347	0.2297
UniEval (Overall)	0.0259	0.1686	0.1814	0.1385
wo-GPT-3.5 (Relevance)	0.0714	-0.1213	0.0464	-0.1533
wo-GPT-3.5 (Consistency)	0.0266	-0.0478	-0.0526	-0.0624
wo-GP-T3.5 (Coherence)	0.0325	-0.0464	-0.0550	-0.0558
wo-GPT-3.5 (Fluency)	0.0105	-0.0094	-0.0039	-0.0272
wo-GPT-3.5 (Overall)	-0.0007	-0.0300	-0.0094	-0.0379
wi-gt-GPT-3.5 (Relevance)	-0.0598	-0.1062	-0.0832	-0.0932
wi-gt-GPT-3.5 (Consistency)	-0.0066	-0.0486	-0.0582	-0.0516
wi-gt-GPT-3.5 (Coherence)	-0.0918	-0.0589	-0.0727	-0.0541
wi-gt-GPT-3.5 (Fluency)	-0.0402	-0.1248	-0.0733	-0.1125
wi-gt-GPT-3.5 (Overall)	-0.0563	-0.0768	-0.0341	-0.0771
wi-in-GPT-3.5 (Relevance)	0.0122	-0.0074	-0.0050	-0.0112
wi-in-GPT-3.5 (Consistency)	-0.0130	-0.0686	-0.0892	-0.0739
wi-in-GPT-3.5 (Coherence)	-0.0295	-0.0117	-0.0832	-0.0214
wi-in-GPT-3.5 (Fluency)	0.0059	-0.1526	-0.1198	-0.1627
wi-in-GPT-3.5 (Overall)	-0.0506	-0.1875	-0.1962	-0.1827
wi-ingt-GPT-3.5 (Relevance)	-0.0152	-0.0612	-0.0431	-0.0502
wi-ingt-GPT-3.5 (Consistency)	-0.0176	-0.0290	-0.0326	-0.0167
wi-ingt-GPT-3.5 (Coherence)	-0.0870	-0.1265	-0.0924	-0.1268
wi-ingt-GPT-3.5 (Fluency)	-0.1197	-0.0844	-0.0136	-0.0609
wi-ingt-GPT-3.5 (Overall)	-0.0136	-0.1040	-0.1022	-0.0979
Col Mean	-0.0109	-0.0142	**-0.0101**	-0.0221

Table 5.6: Main results of the relationship between the uncertainty estimation methods and NLG metrics on XSUM dataset using generation from GPT-3.5.

NLG Metrics	**MSP**	**MTE**	**MCSE**	**MD**	**RDE**	**T-TU**	**T-RMI**	**S-TU**	**S-RMI**
ROUGE-L	0.1763	-0.0385	0.1539	-0.0333	-0.0844	-0.0571	-0.0752	-0.0849	0.0577
BARTSCORE	-0.0541	0.0314	0.0252	0.1228	0.0930	0.0525	-0.0867	0.0026	0.0590
SummaC	0.0106	-0.0024	0.0304	0.0215	-0.0719	-0.0247	-0.0422	0.0229	0.0267
CTC	0.1101	-0.2225	0.0569	-0.2217	-0.2708	-0.1256	-0.0538	-0.1675	0.0765
Spearman	0.0581	0.0108	0.1008	0.0358	-0.0096	0.0393	-0.0444	-0.0038	0.0472
Kendall-Tau	0.0659	0.0080	0.1043	0.0316	-0.0141	0.0345	-0.0443	-0.0077	0.0458
UniEval (Relevance)	-0.0262	0.1157	0.0480	0.0851	0.0233	0.1165	0.1175	0.1388	0.1046
UniEval (Consistency)	0.3493	0.4330	0.2544	0.3288	0.2752	0.2014	0.0470	0.2281	0.1353
UniEval (Coherence)	0.4194	0.5342	0.2205	0.4139	0.3656	0.2693	0.0839	0.3005	0.0952
UniEval (Fluency)	0.1141	0.0538	0.0782	-0.0489	-0.0556	0.0228	0.1079	0.0917	0.0844
UniEval (Overall)	0.1508	0.2423	0.1274	0.1586	0.1071	0.1490	0.1154	0.1902	0.1172
wo-GPT-3.5 (Relevance)	-0.0183	0.1433	0.0632	0.1613	0.0862	0.0948	-0.1168	0.0700	0.0527
wo-GPT-3.5 (Consistency)	0.1177	0.1945	0.0912	0.1519	0.1255	0.0422	-0.0873	0.0435	0.0546
wo-GP-T3.5 (Coherence)	0.2006	0.2308	0.0798	0.1344	0.1176	0.0535	-0.0777	0.0568	0.0143
wo-GPT-3.5 (Fluency)	0.1277	0.2753	0.0995	0.2308	0.1792	0.0913	-0.0589	0.1319	0.0489
wo-GPT-3.5 (Overall)	0.0287	0.2327	0.0661	0.1784	0.1434	0.1107	-0.1695	0.0788	0.0688
Col Mean	0.1144	**0.1402**	0.1000	0.1094	0.0631	0.0669	-0.0241	0.0682	0.0681

Table 5.7: Main results of the relationship between the uncertainty estimation methods and NLG metrics on AESLC dataset using generation from BART.

NLG Metrics	**MSP**	**MTE**	**MCSE**	**MD**	**RDE**	**T-TU**	**T-RMI**	**S-TU**	**S-RMI**
ROUGE-L	-0.0978	-0.1359	-0.0587	-0.0375	-0.0996	-0.0511	-0.0126	-0.0733	-0.0041
BARTSCORE	-0.0558	0.0009	0.0752	0.1515	0.1324	0.0037	0.1136	-0.0457	-0.0040
SummaC	0.0760	0.0148	0.0696	-0.0004	-0.0542	-0.0432	0.0611	-0.0227	-0.0144
CTC	-0.0486	-0.3754	-0.0001	-0.2301	-0.3298	-0.1879	0.0176	-0.2027	-0.0095
Spearman	-0.2040	-0.1310	-0.1209	0.0028	-0.0424	-0.0579	-0.0500	-0.0895	-0.0048
Kendall-Tau	-0.2013	-0.1321	-0.1206	-0.0005	-0.0452	-0.0586	-0.0511	-0.0906	-0.0048
UniEval (Relevance)	-0.1475	-0.1092	-0.0562	-0.0126	-0.0966	-0.0858	0.0156	-0.1038	-0.0235
UniEval (Consistency)	0.7162	0.7064	0.6192	0.5025	0.5037	0.1936	0.1523	0.2284	0.0223
UniEval (Coherence)	0.6467	0.6474	0.5416	0.4556	0.4740	0.1358	0.1423	0.1790	0.0095
UniEval (Fluency)	0.0345	-0.1334	0.0338	0.0493	0.0469	-0.1619	0.1130	-0.2010	0.0375
UniEval (Overall)	0.2380	0.2421	0.2456	0.2264	0.1792	0.0076	0.0920	0.0113	-0.0033
wo-GPT-3.5 (Relevance)	0.0667	0.1933	0.0761	0.1921	0.1629	0.0292	-0.0060	0.0261	0.0081
wo-GPT-3.5 (Consistency)	0.1220	0.1545	0.0544	0.0828	0.0777	0.0235	-0.0382	0.0299	0.0051
wo-GP-T3.5 (Coherence)	0.1436	0.2104	0.0479	0.1176	0.1313	0.0454	-0.0740	0.0527	0.0046
wo-GPT-3.5 (Fluency)	0.1924	0.2679	0.0675	0.1547	0.1815	0.0769	-0.0528	0.0816	0.0102
wo-GPT-3.5 (Overall)	0.1137	0.2039	0.0448	0.1446	0.1546	0.0848	-0.1038	0.0681	0.0115
Col Mean	0.0997	0.1015	0.0950	**0.1124**	0.0860	-0.0029	0.0199	-0.0095	0.0025

Table 5.8: Main results of the relationship between the uncertainty estimation methods and NLG metrics on XSUM dataset using generation from BART.

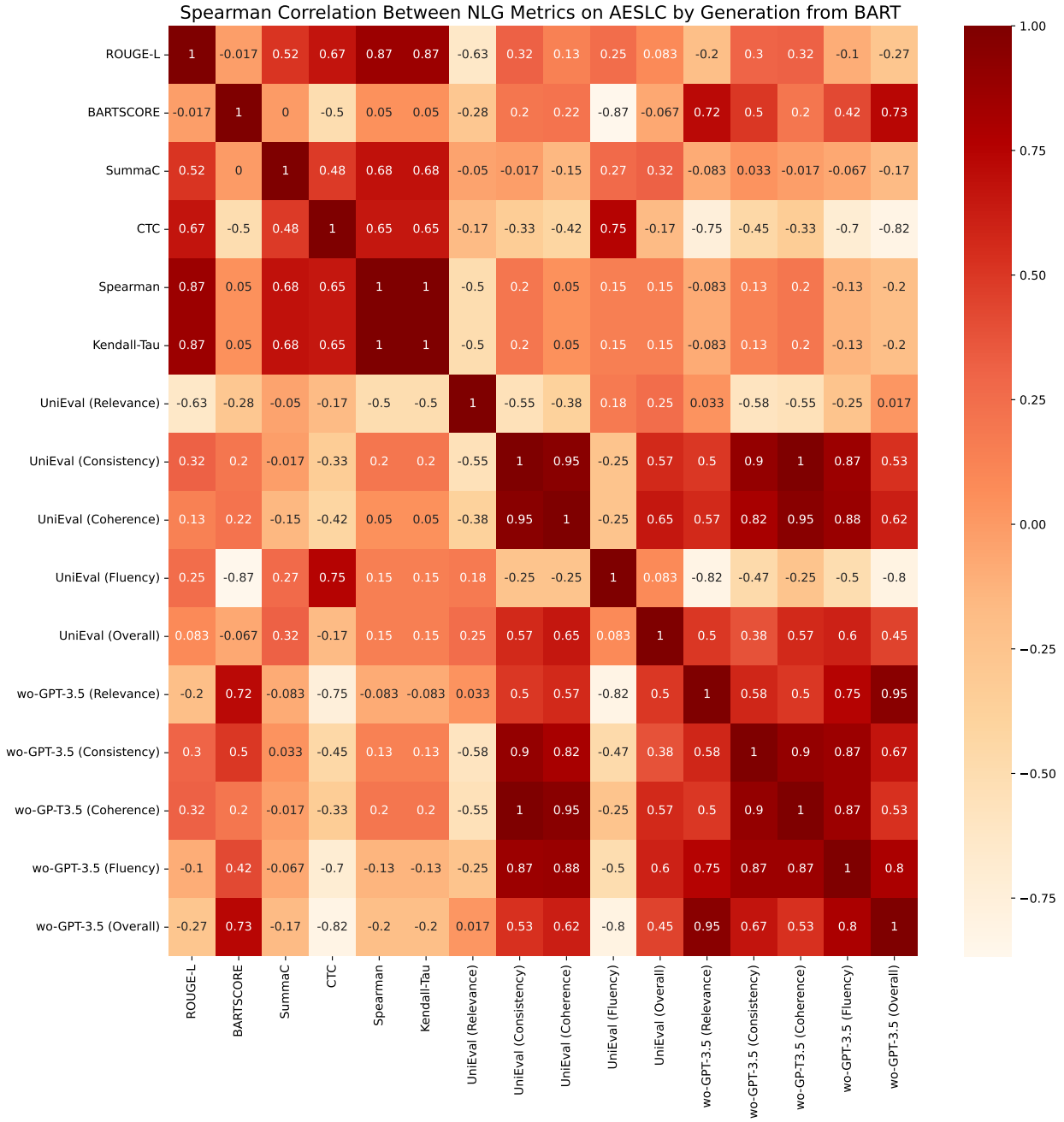

Figure 5.3: Diagram of Spearman correlation between NLG metrics on AESLC dataset from the view of uncertainty estimation methods used in Fig. 5.6. The generated summaries are from BART. For the GPT-3.5-based NLG metrics, we only conduct wo-GPT-3.5 on the BART generation model setting.

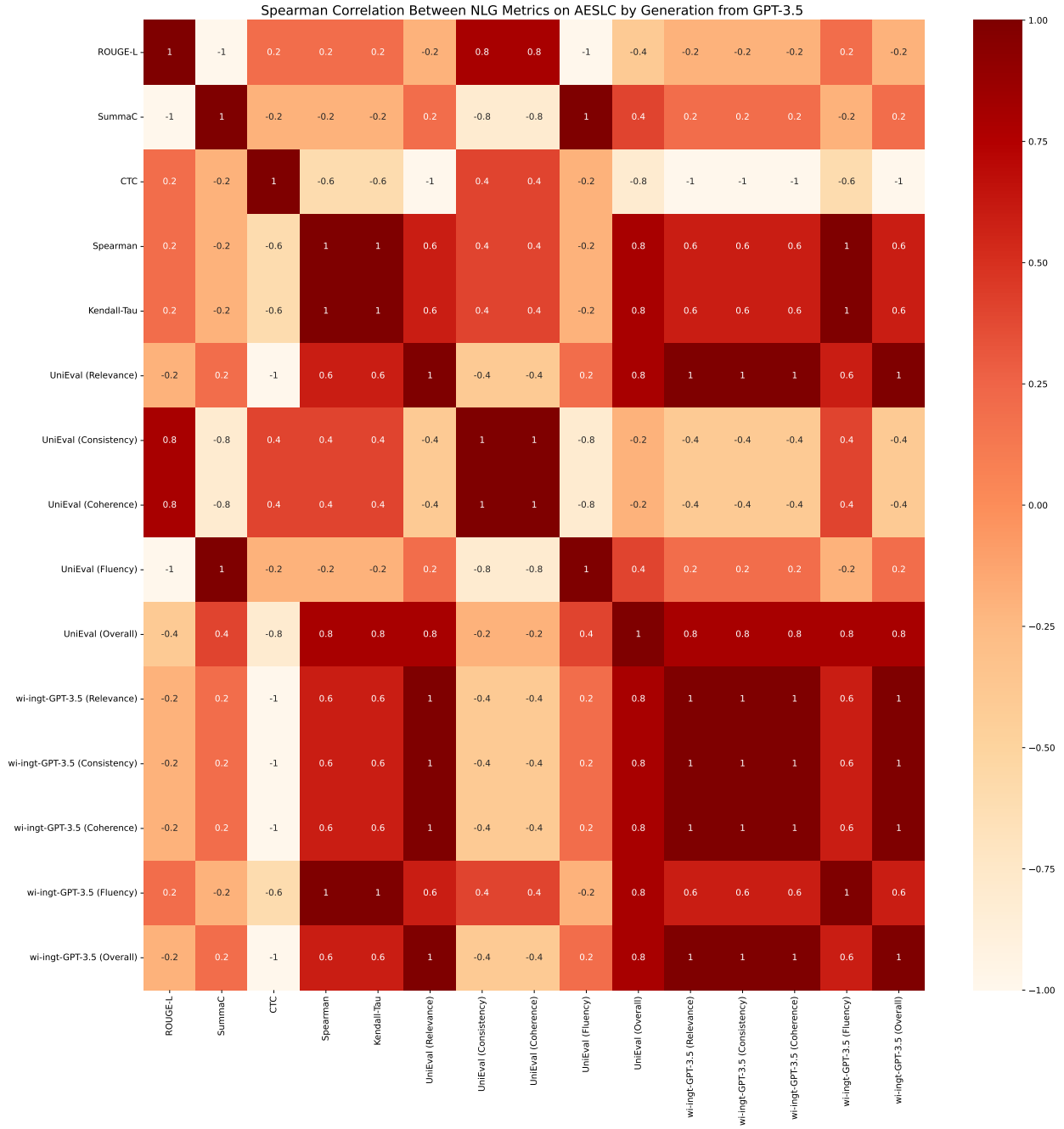

Figure 5.4: Diagram of Spearman correlation between NLG metrics on AESLC dataset from the view of uncertainty estimation methods used in Fig. 5.7. The generated summaries are from GPT-3.5. For the GPT-3.5-based NLG metrics, we only draw wi-ingt-GPT-3.5 results to save space.

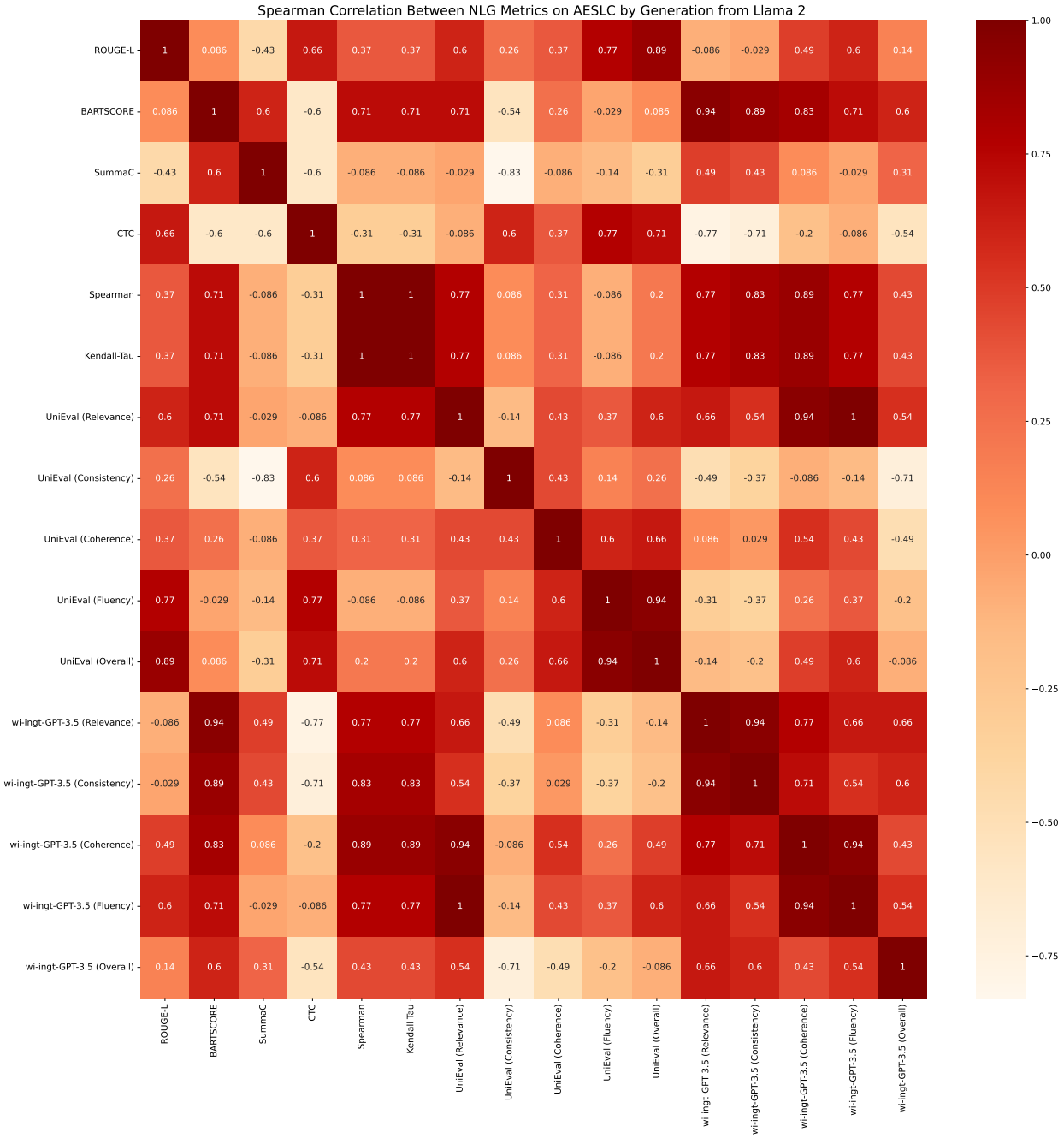

Figure 5.5: Diagram of Spearman correlation between NLG metrics on AESLC dataset from the view of uncertainty estimation methods used in Fig. 5.8. The generated summaries are from Llama 2. For the GPT-3.5-based NLG metrics, we only draw wi-ingt-GPT-3.5 results to save space.

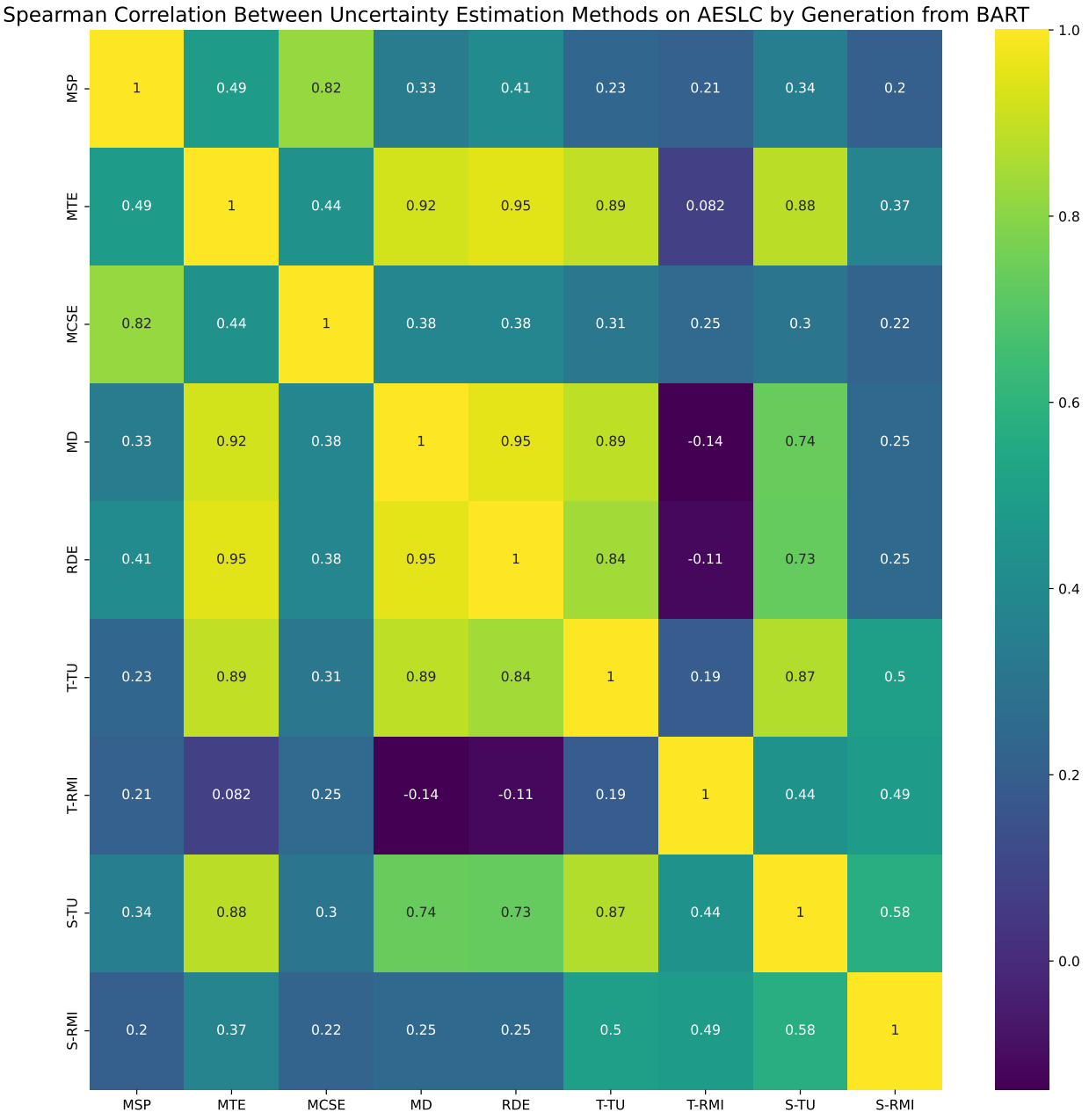

Figure 5.6: Diagram of Spearman correlation between uncertainty estimation methods on AESLC dataset from the view of NLG metrics used in Fig. 5.3. The generated summaries are from BART.

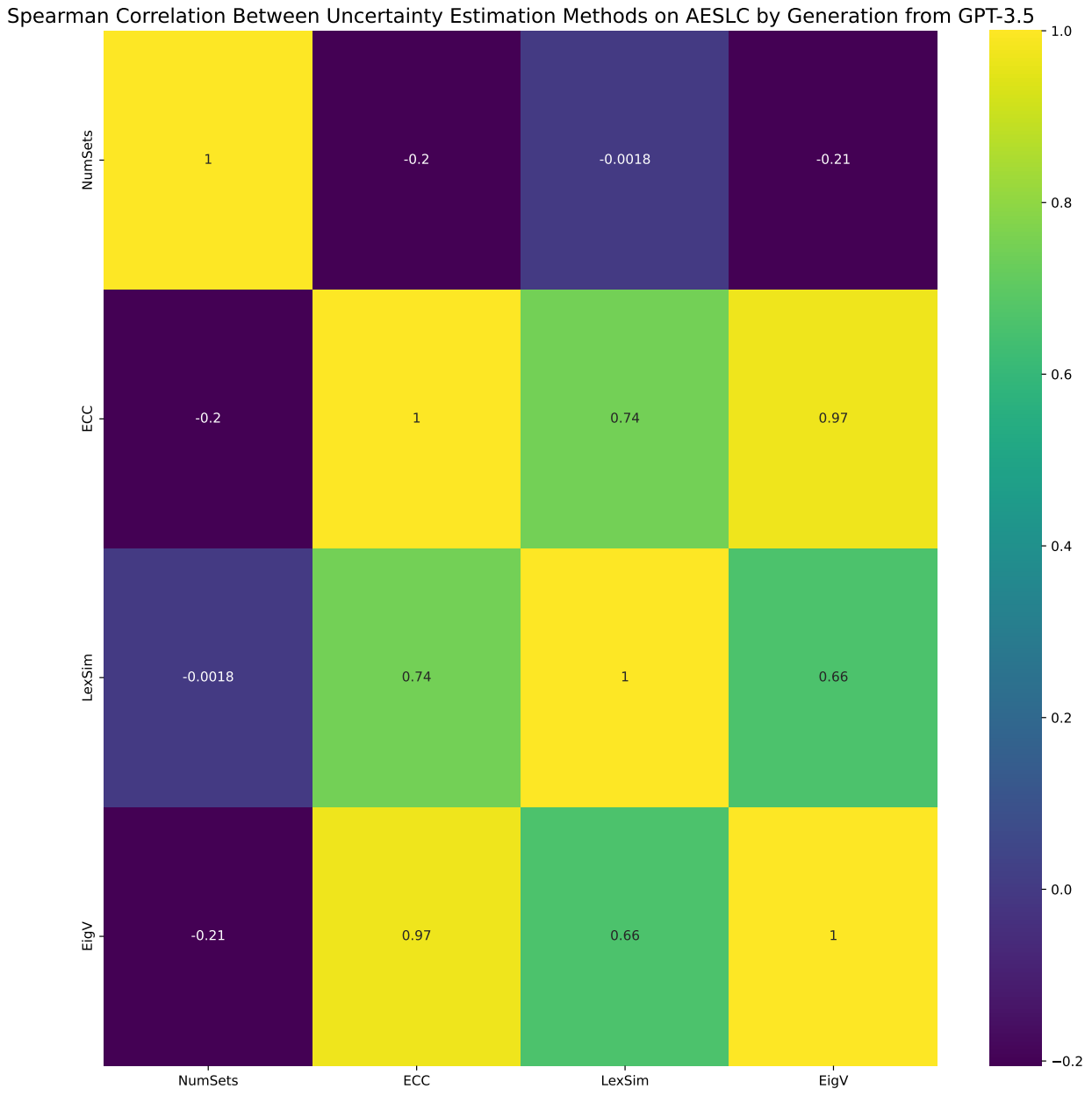

Figure 5.7: Diagram of Spearman correlation between uncertainty estimation methods on AESLC dataset from the view of NLG metrics used in Fig. 5.4. The generated summaries are from GPT-3.5.

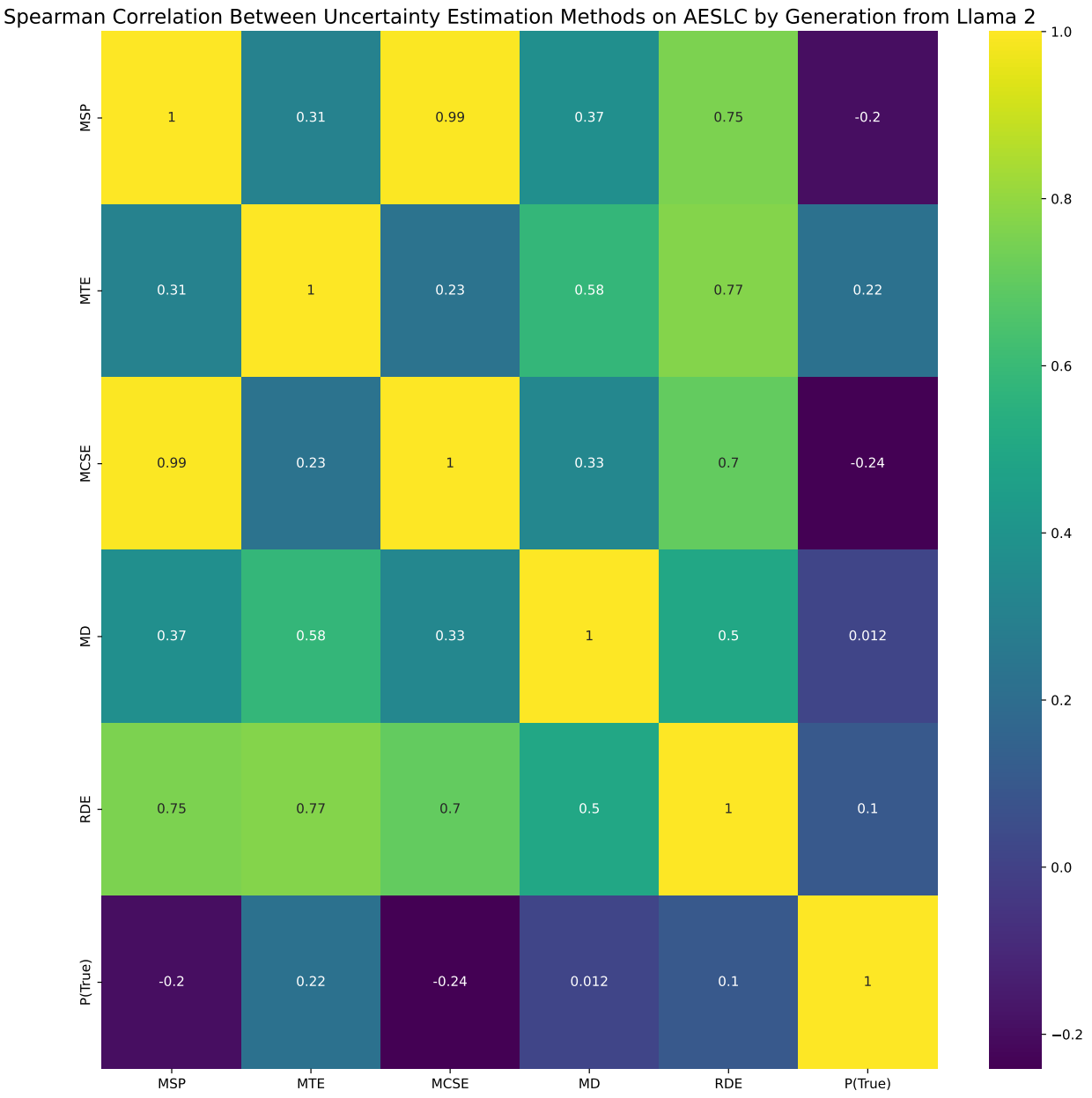

Figure 5.8: Diagram of Spearman correlation between uncertainty estimation methods on AESLC dataset from the view of NLG metrics used in Fig. 5.5. The generated summaries are from Llama 2.

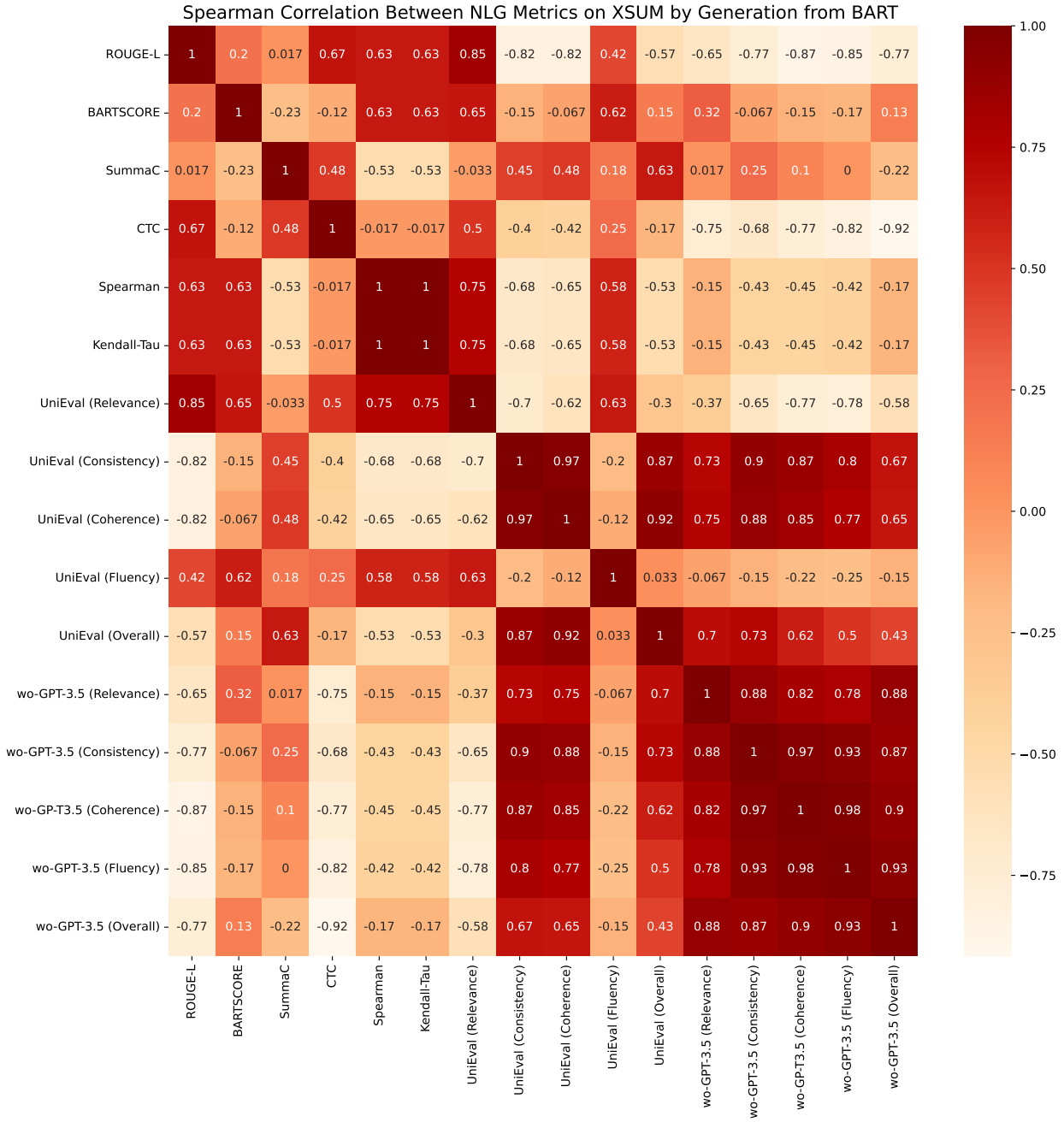

Figure 5.9: Diagram of Spearman correlation between NLG metrics on XSUM dataset from the view of uncertainty estimation methods used in Fig. 5.12. The generated summaries are from BART. For the GPT-3.5-based NLG metrics, we only conduct wo-GPT-3.5 on the BART generation model setting.

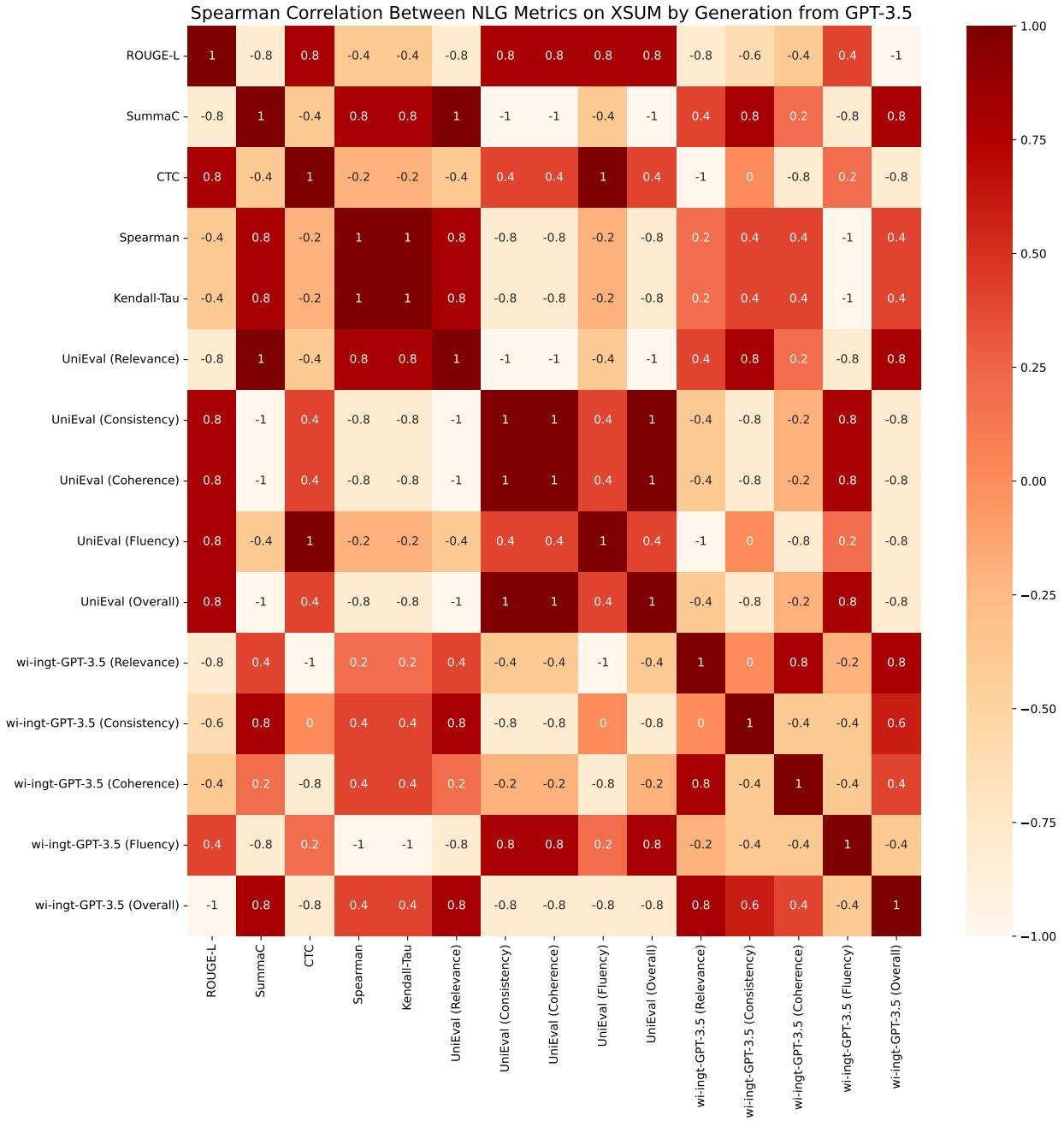

Figure 5.10: Diagram of Spearman correlation between NLG metrics on XSUM dataset from the view of uncertainty estimation methods used in Fig. 5.13. The generated summaries are from GPT-3.5. For the GPT-3.5-based NLG metrics, we only draw wi-ingt-GPT-3.5 results to save space.

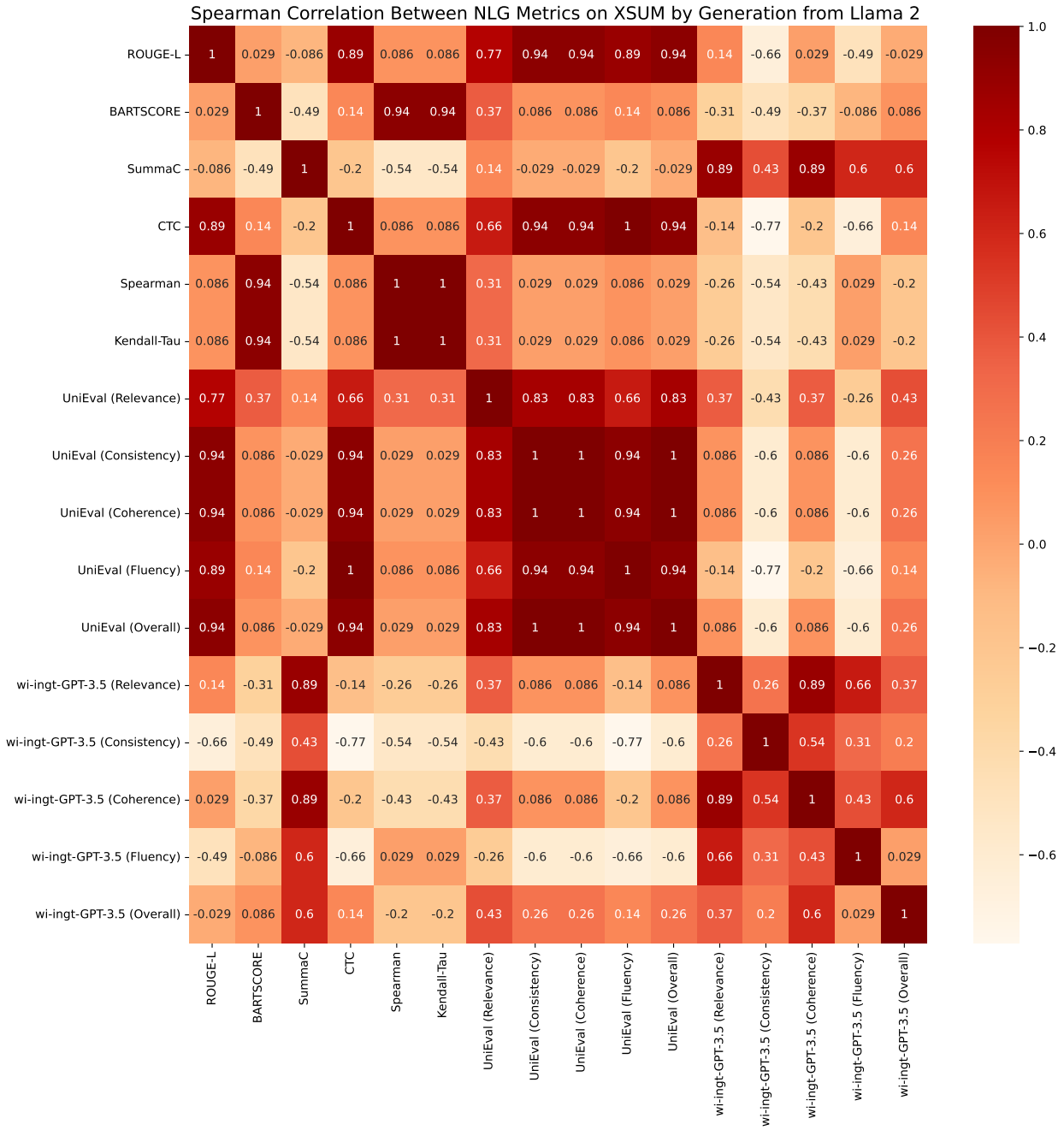

Figure 5.11: Diagram of Spearman correlation between NLG metrics on XSUM dataset from the view of uncertainty estimation methods used in Fig. 5.14. The generated summaries are from Llama 2. For the GPT-3.5-based NLG metrics, we only draw wi-ingt-GPT-3.5 results to save space.

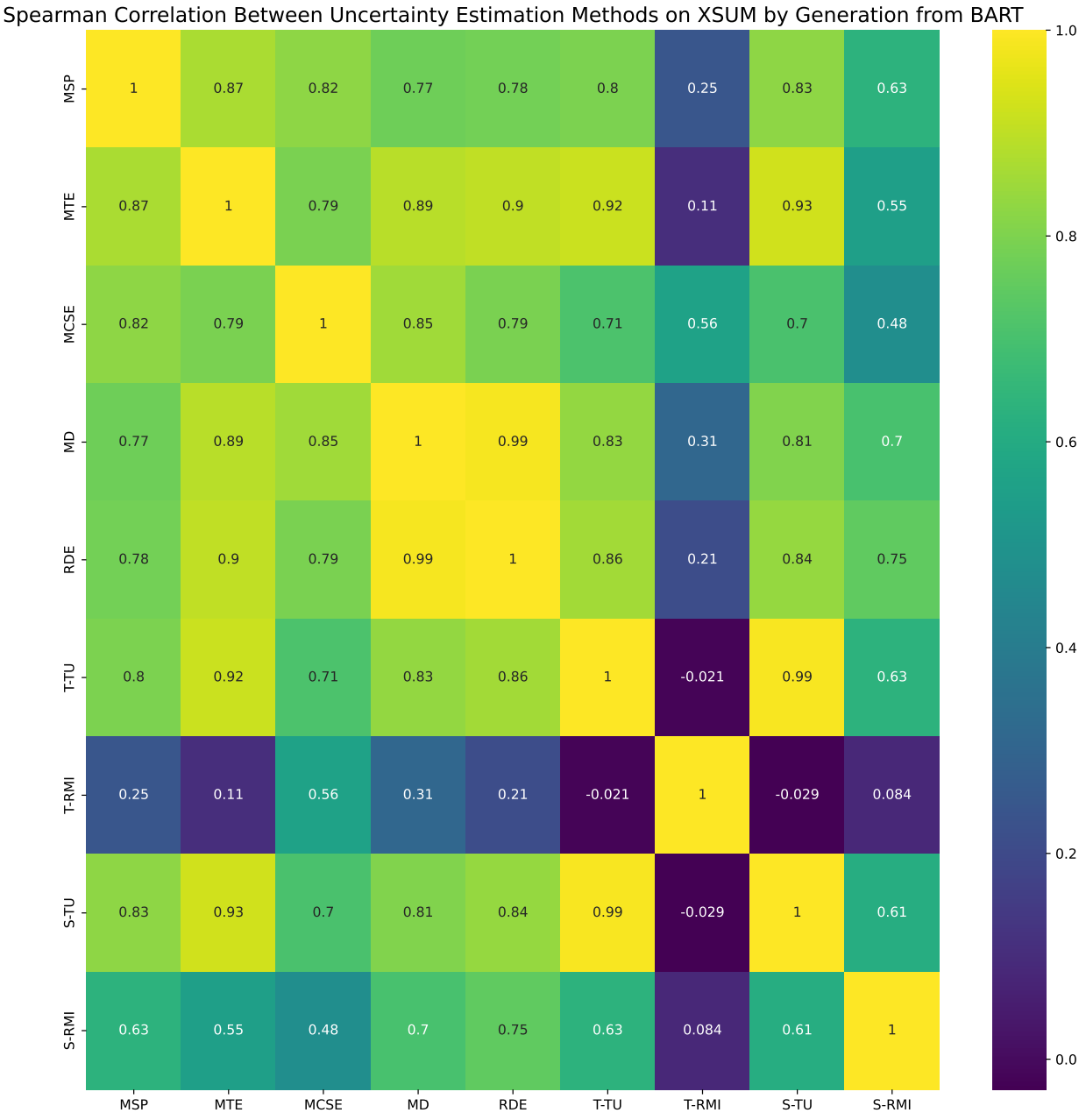

Figure 5.12: Diagram of Spearman correlation between uncertainty estimation methods on XSUM dataset from the view of NLG metrics used in Fig. 5.9. The generated summaries are from BART.

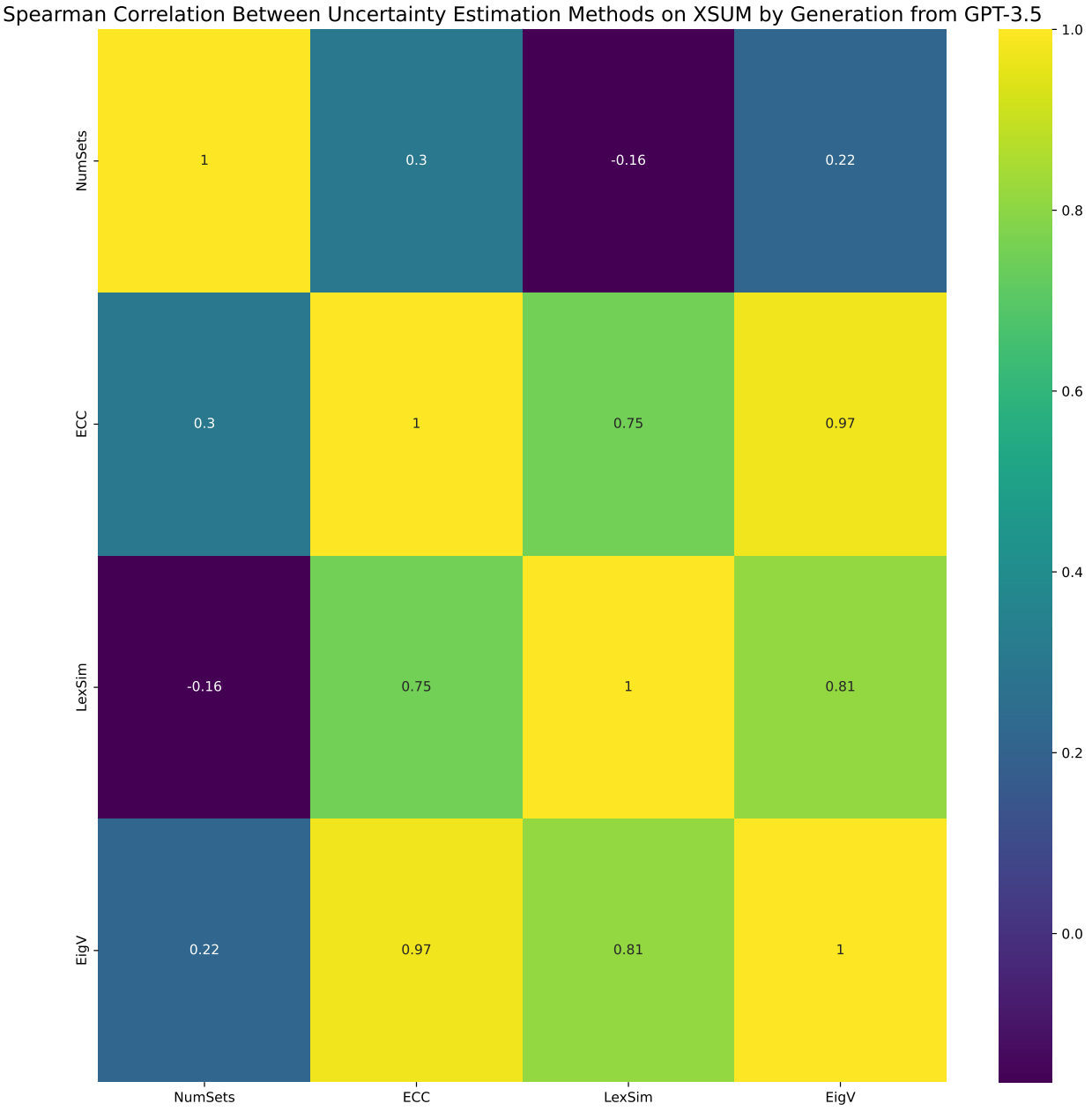

Figure 5.13: Diagram of Spearman correlation between uncertainty estimation methods on XSUM dataset from the view of NLG metrics used in Fig. 5.10. The generated summaries are from GPT-3.5.

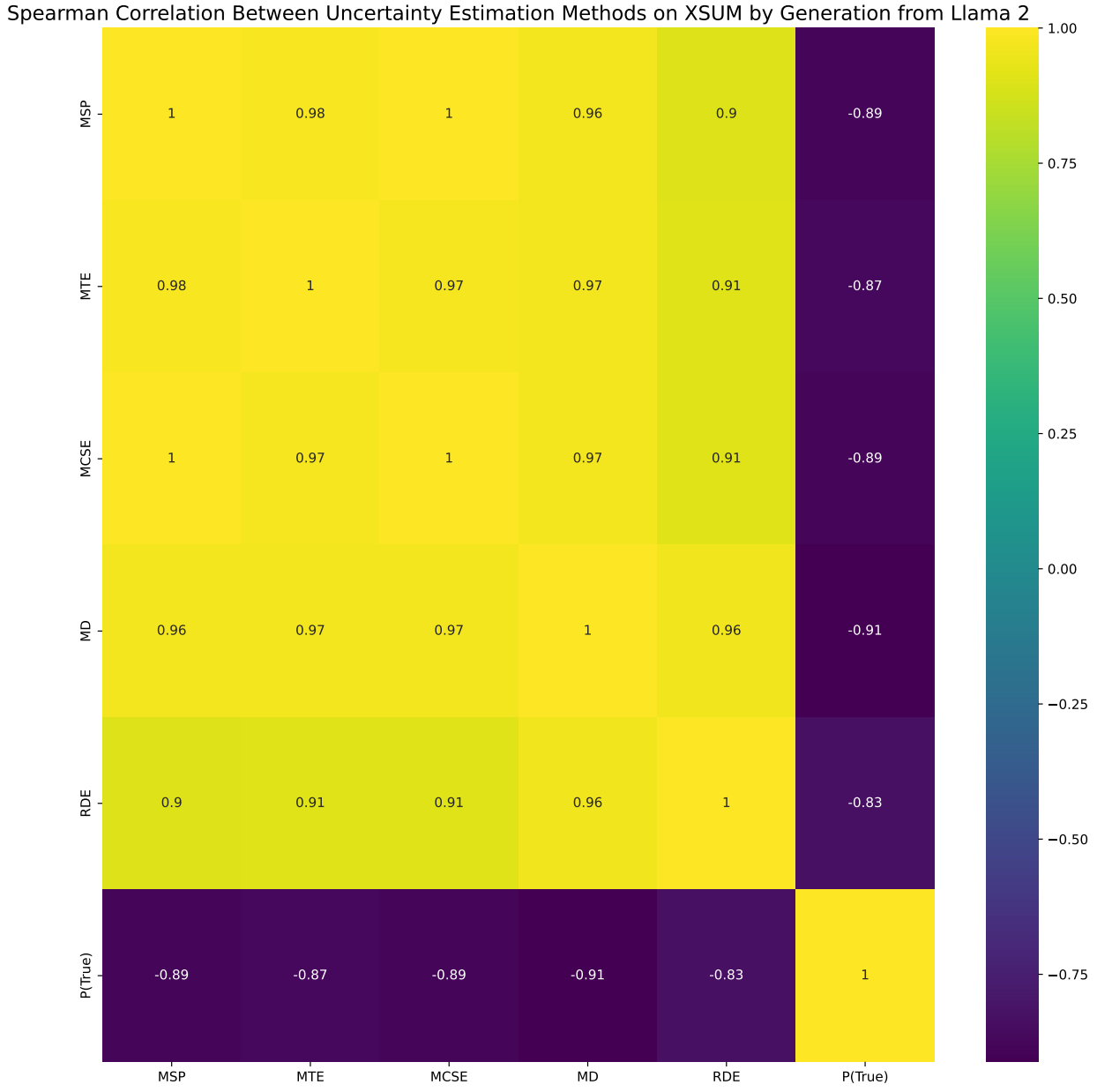

Figure 5.14: Diagram of Spearman correlation between uncertainty estimation methods on XSUM dataset from the view of NLG metrics used in Fig. 5.11. The generated summaries are from Llama 2.

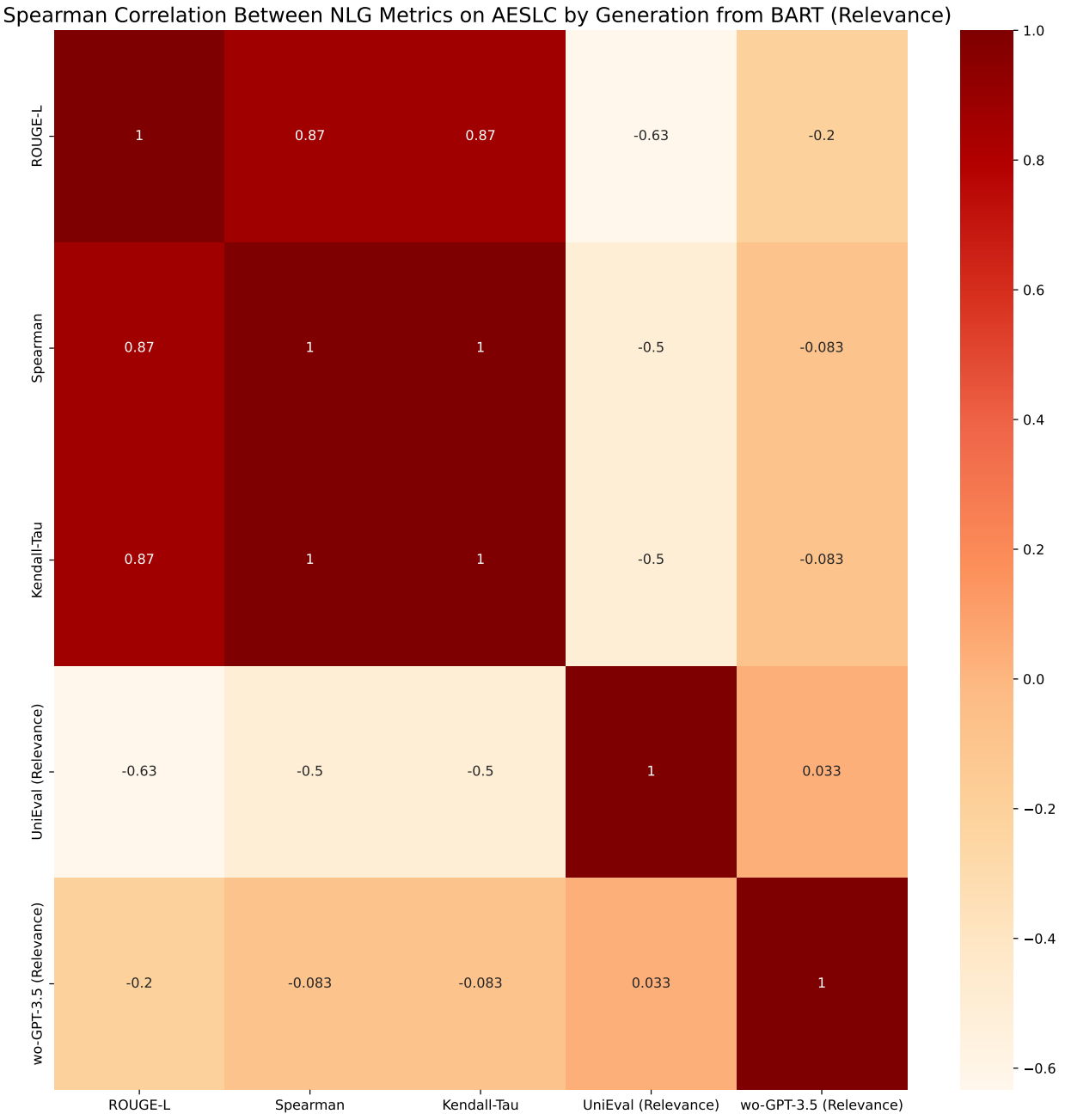

Figure 5.15: Diagram of Spearman correlation in terms of relevance between NLG metrics on AESLC dataset from the view of uncertainty estimation methods used in Fig. 5.6. The generated summaries are from BART. For the GPT-3.5-based NLG metrics, we only conduct wo-GPT-3.5 on the BART generation model setting.

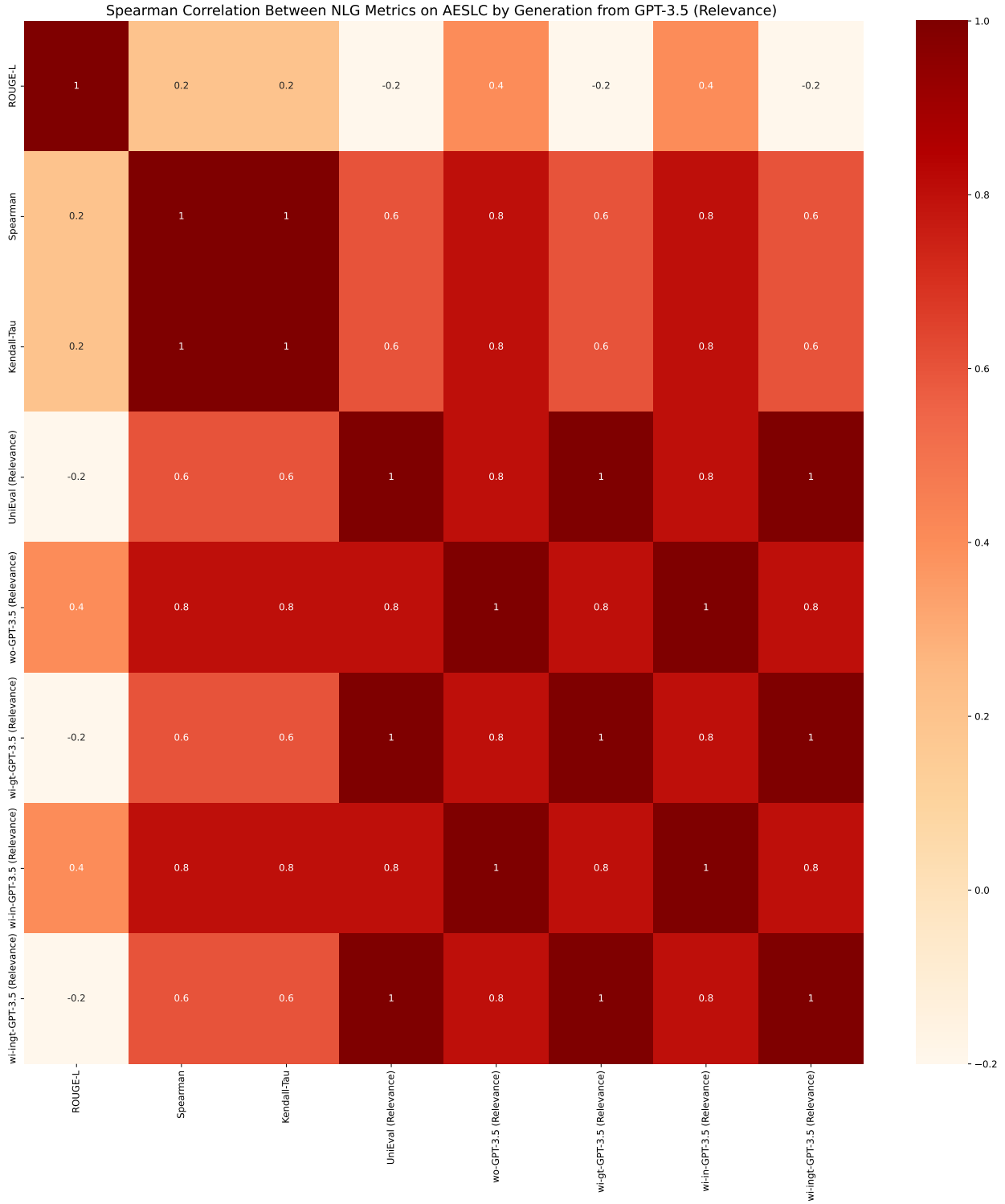

Figure 5.16: Diagram of Spearman correlation in terms of relevance between NLG metrics on AESLC dataset from the view of uncertainty estimation methods used in Fig. 5.7. The generated summaries are from GPT-3.5. For the GPT-3.5-based NLG metrics, we only draw wi-ingt-GPT-3.5 results to save space.

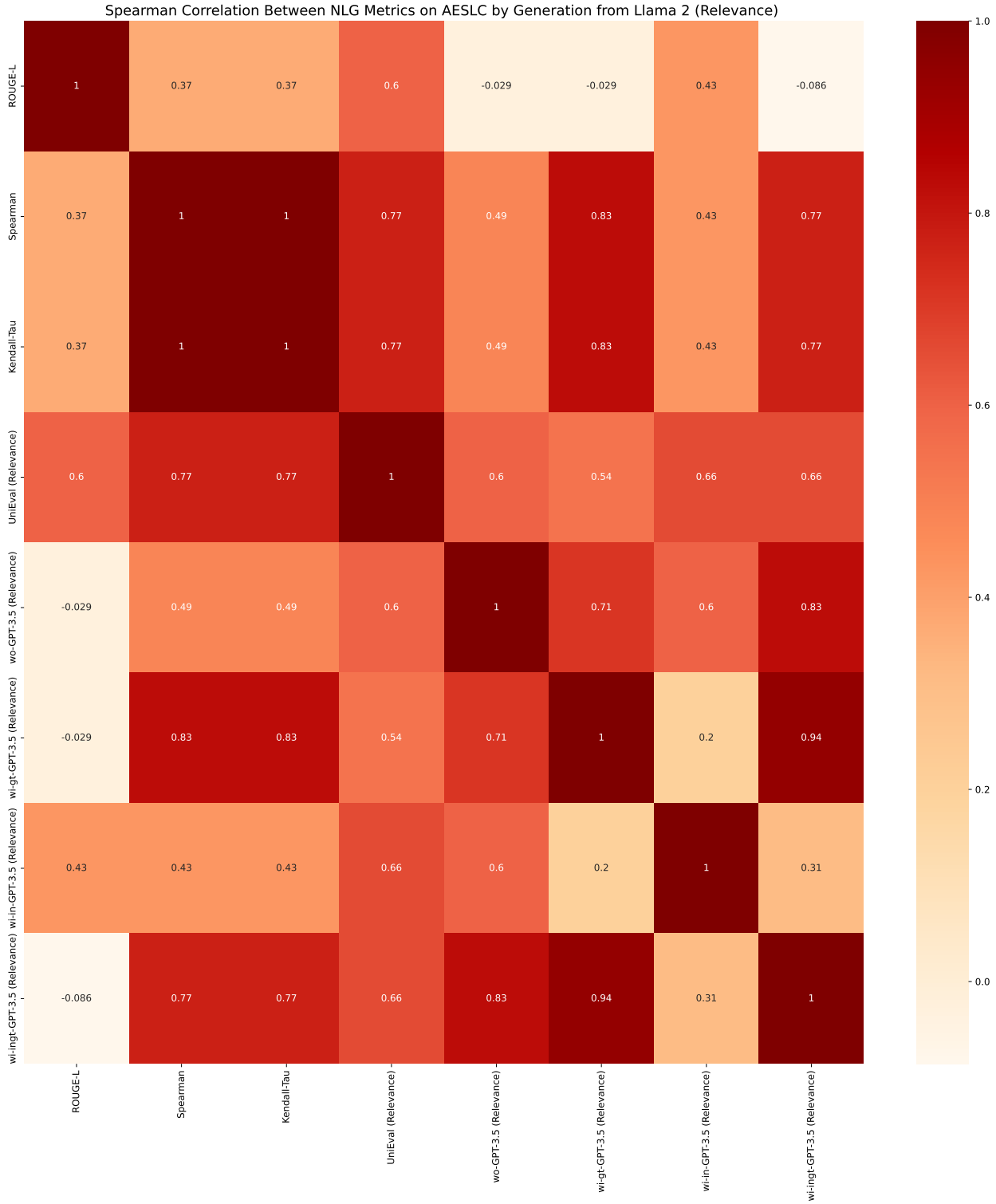

Figure 5.17: Diagram of Spearman correlation in terms of relevance between NLG metrics on AESLC dataset from the view of uncertainty estimation methods used in Fig. 5.8. The generated summaries are from Llama 2. For the GPT-3.5-based NLG metrics, we only draw wi-ingt-GPT-3.5 results to save space.

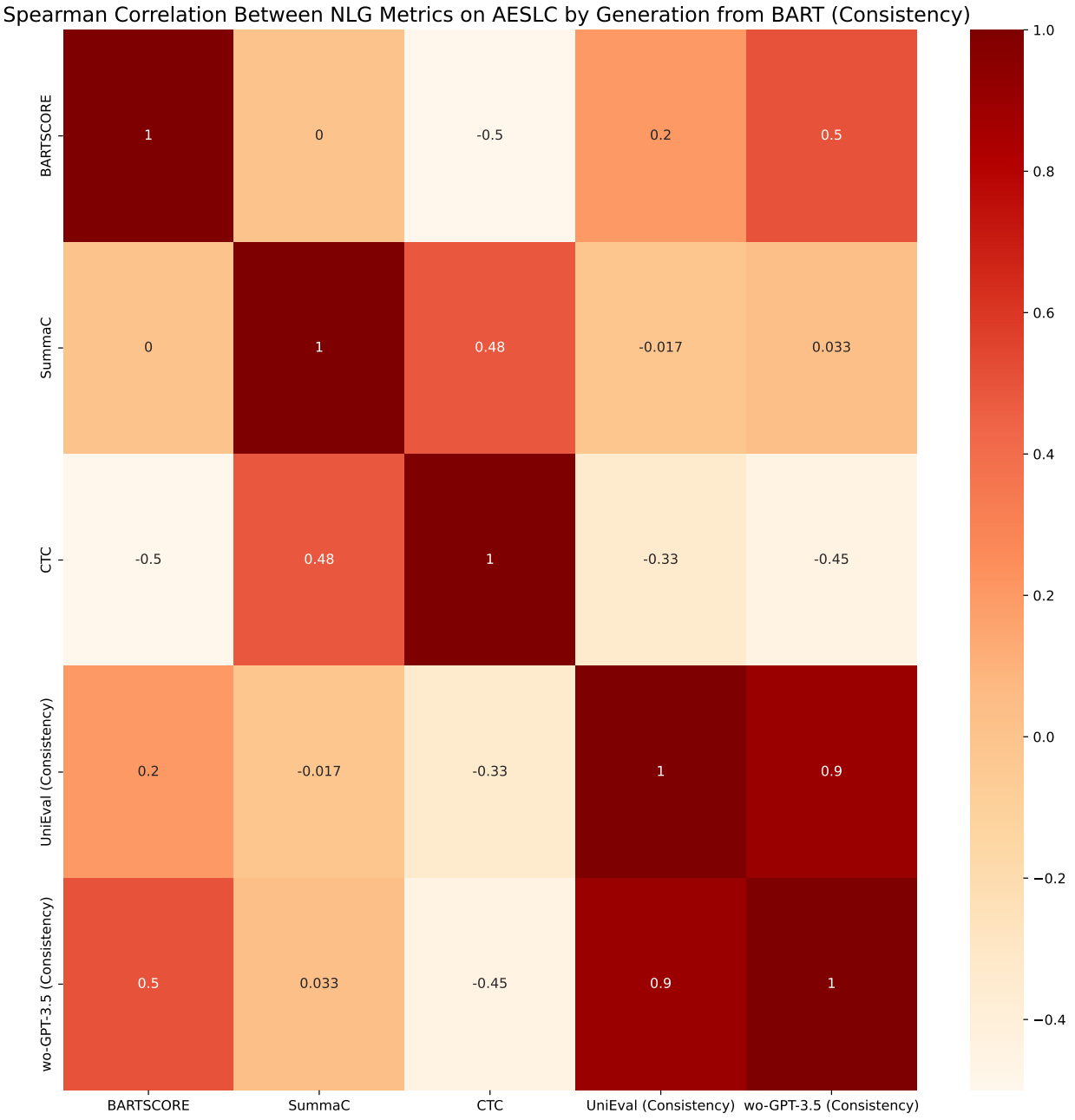

Figure 5.18: Diagram of Spearman correlation in terms of consistency between NLG metrics on AESLC dataset from the view of uncertainty estimation methods used in Fig. 5.6. The generated summaries are from BART. For the GPT-3.5-based NLG metrics, we only conduct wo-GPT-3.5 on the BART generation model setting.

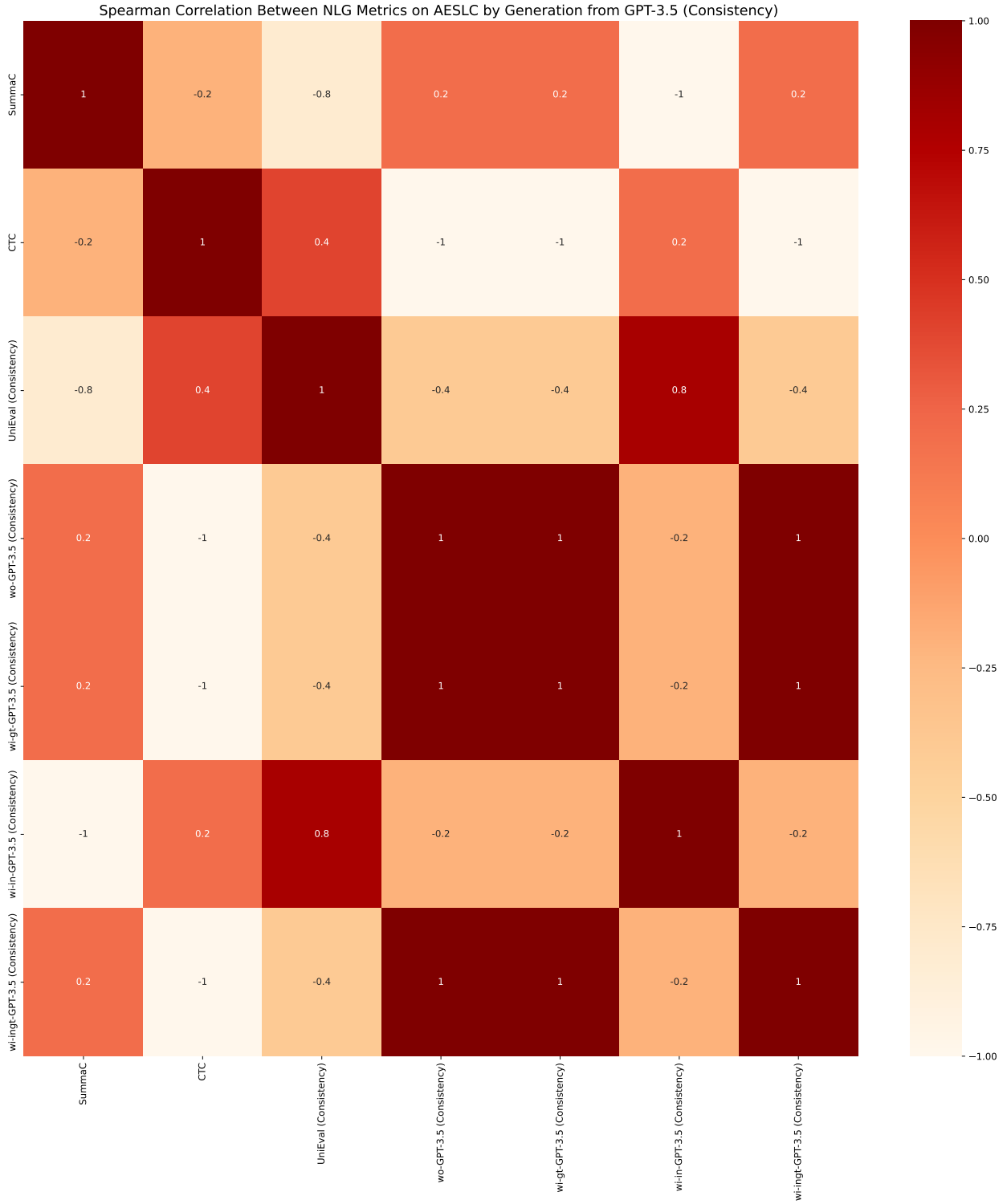

Figure 5.19: Diagram of Spearman correlation in terms of consistency between NLG metrics on AESLC dataset from the view of uncertainty estimation methods used in Fig. 5.7. The generated summaries are from GPT-3.5. For the GPT-3.5-based NLG metrics, we only draw wi-ingt-GPT-3.5 results to save space.

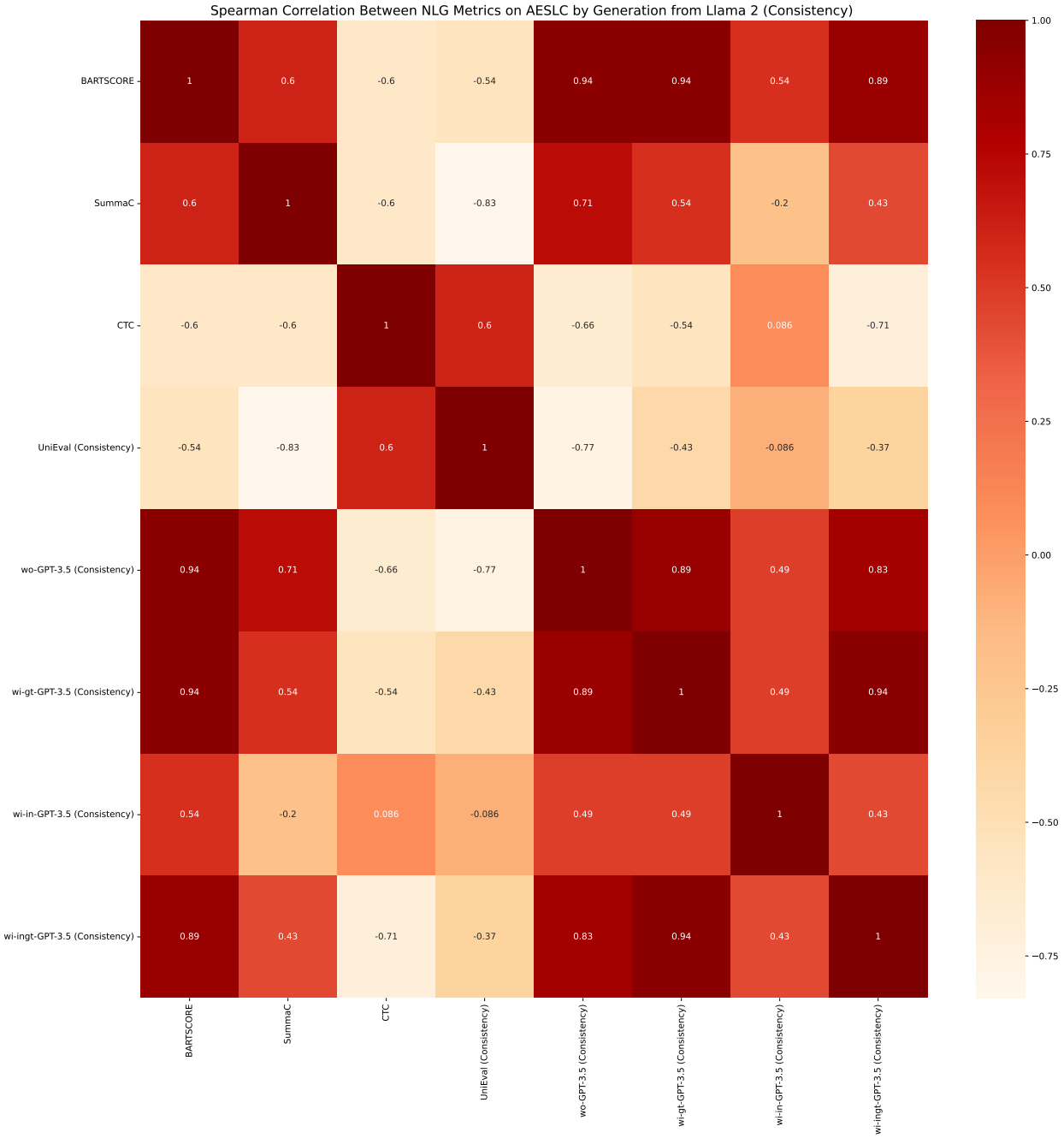

Figure 5.20: Diagram of Spearman correlation in terms of consistency between NLG metrics on AESLC dataset from the view of uncertainty estimation methods used in Fig. 5.8. The generated summaries are from Llama 2. For the GPT-3.5-based NLG metrics, we only draw wi-ingt-GPT-3.5 results to save space.

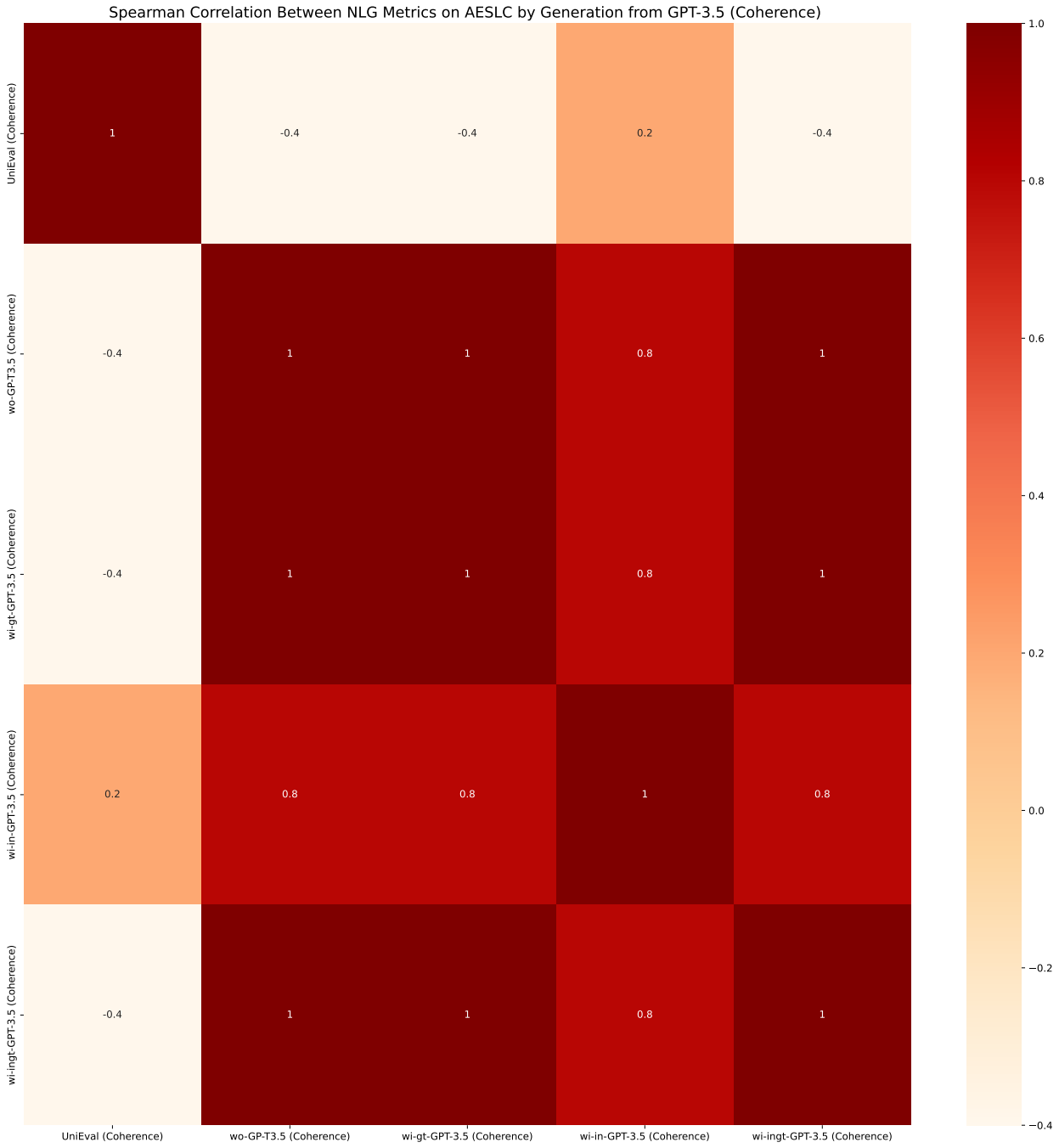

Figure 5.21: Diagram of Spearman correlation in terms of coherence between NLG metrics on AESLC dataset from the view of uncertainty estimation methods used in Fig. 5.7. The generated summaries are from GPT-3.5. For the GPT-3.5-based NLG metrics, we only draw wi-ingt-GPT-3.5 results to save space.

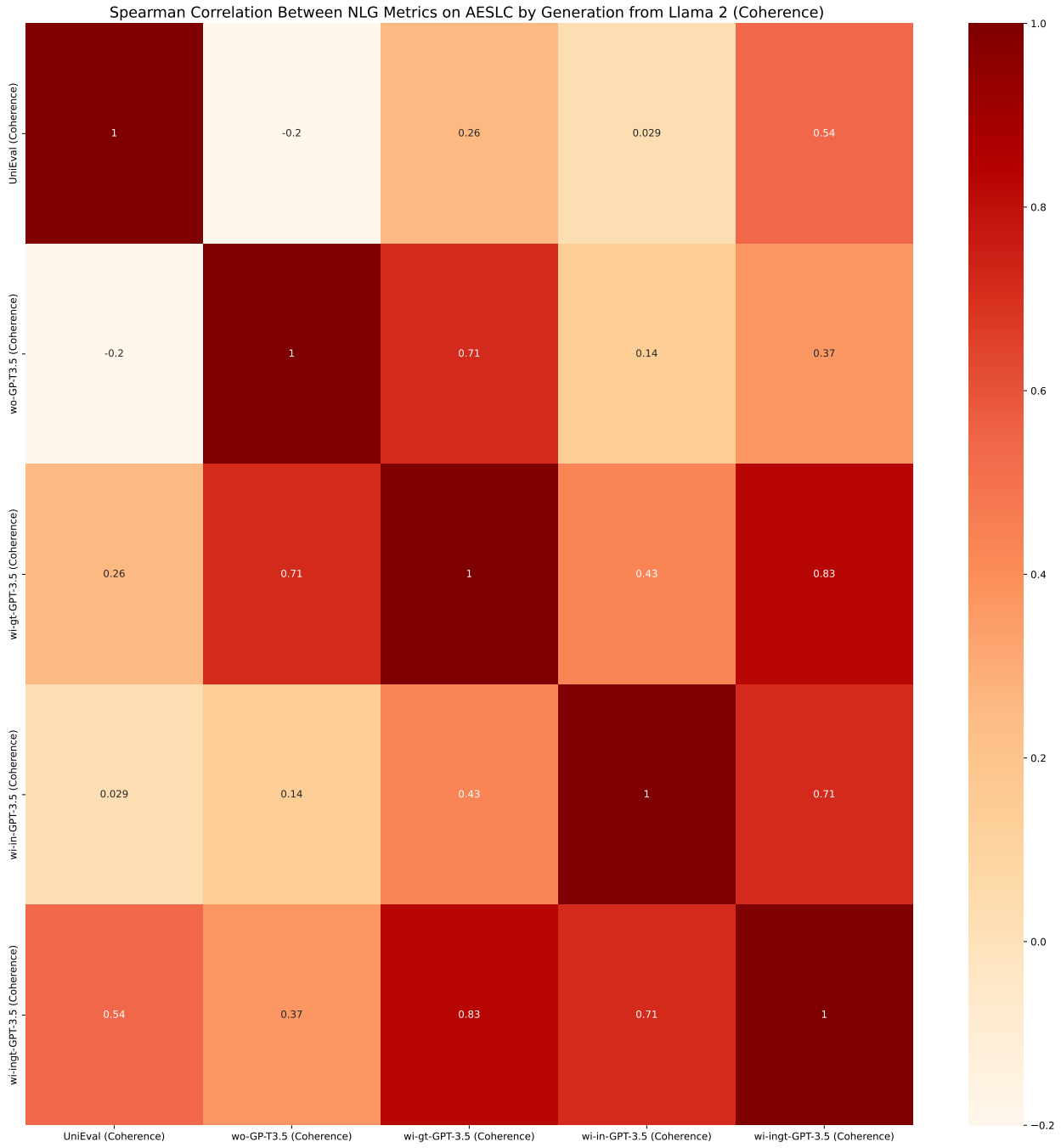

Figure 5.22: Diagram of Spearman correlation in terms of coherence between NLG metrics on AESLC dataset from the view of uncertainty estimation methods used in Fig. 5.8. The generated summaries are from Llama 2. For the GPT-3.5-based NLG metrics, we only draw wi-ingt-GPT-3.5 results to save space.

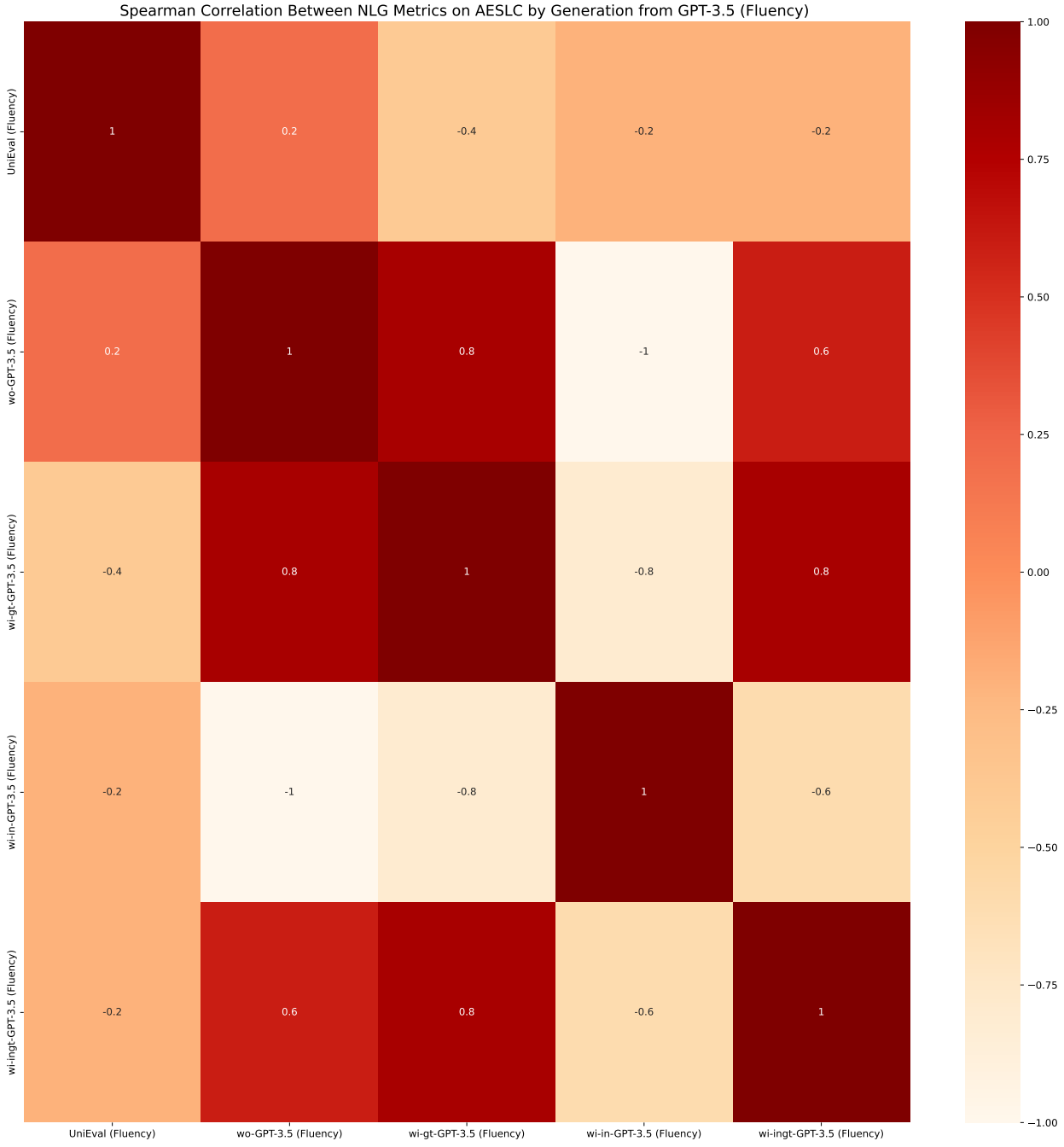

Figure 5.23: Diagram of Spearman correlation in terms of fluency between NLG metrics on AESLC dataset from the view of uncertainty estimation methods used in Fig. 5.7. The generated summaries are from GPT-3.5. For the GPT-3.5-based NLG metrics, we only draw wi-ingt-GPT-3.5 results to save space.

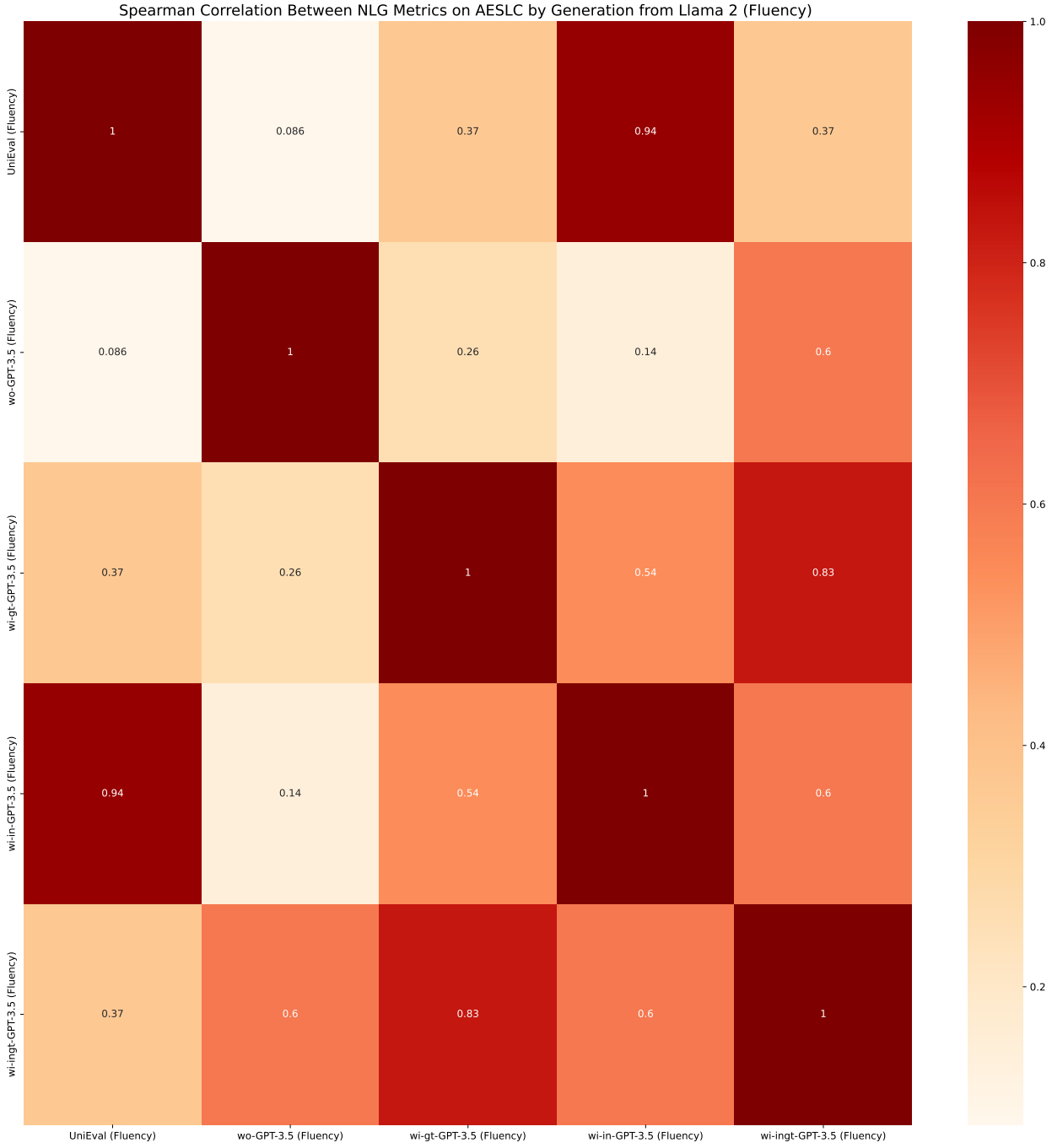

Figure 5.24: Diagram of Spearman correlation in terms of fluency between NLG metrics on AESLC dataset from the view of uncertainty estimation methods used in Fig. 5.8. The generated summaries are from Llama 2. For the GPT-3.5-based NLG metrics, we only draw wi-ingt-GPT-3.5 results to save space.

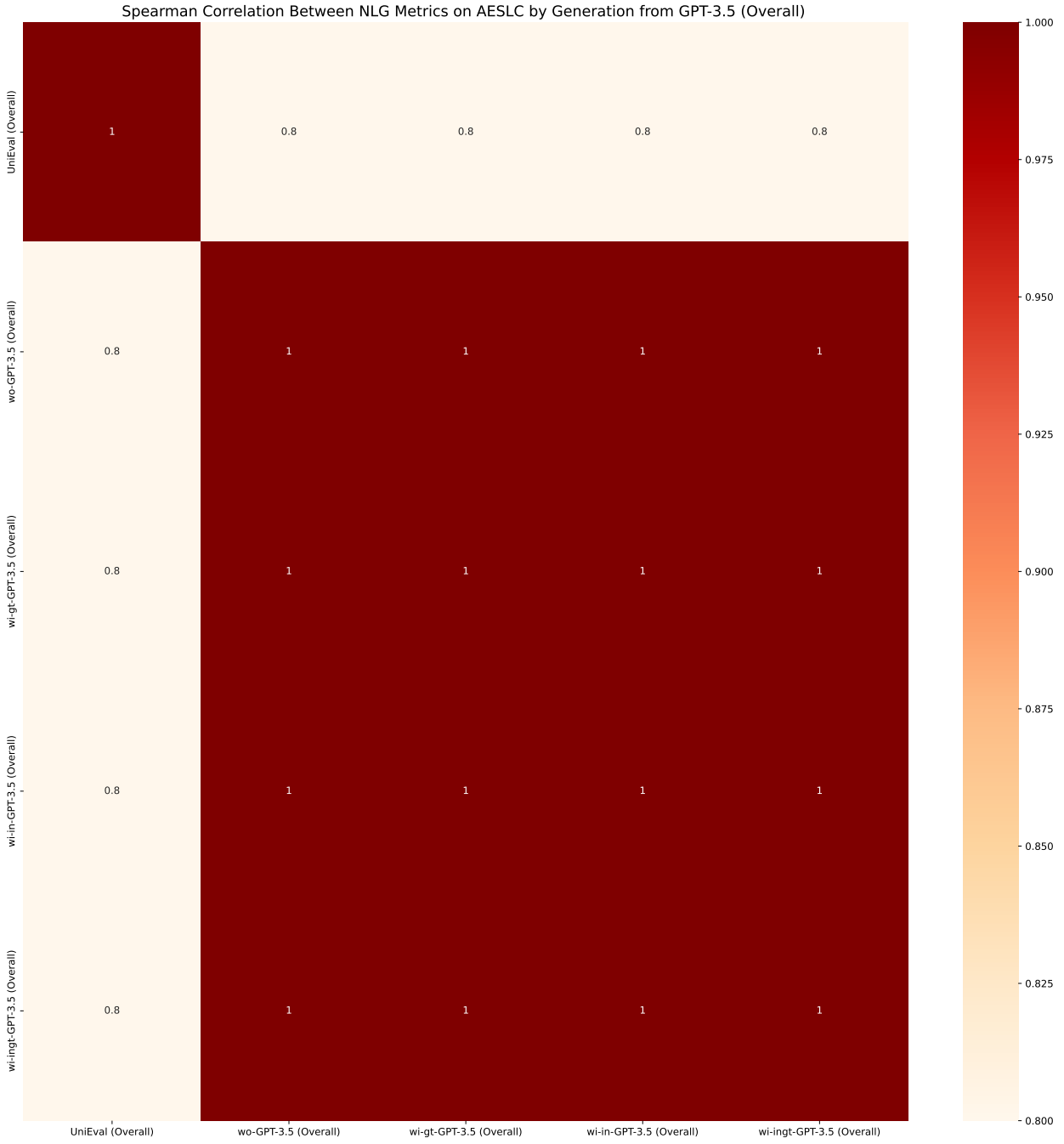

Figure 5.25: Diagram of Spearman correlation in terms of overall between NLG metrics on AESLC dataset from the view of uncertainty estimation methods used in Fig. 5.7. The generated summaries are from GPT-3.5. For the GPT-3.5-based NLG metrics, we only draw wi-ingt-GPT-3.5 results to save space.

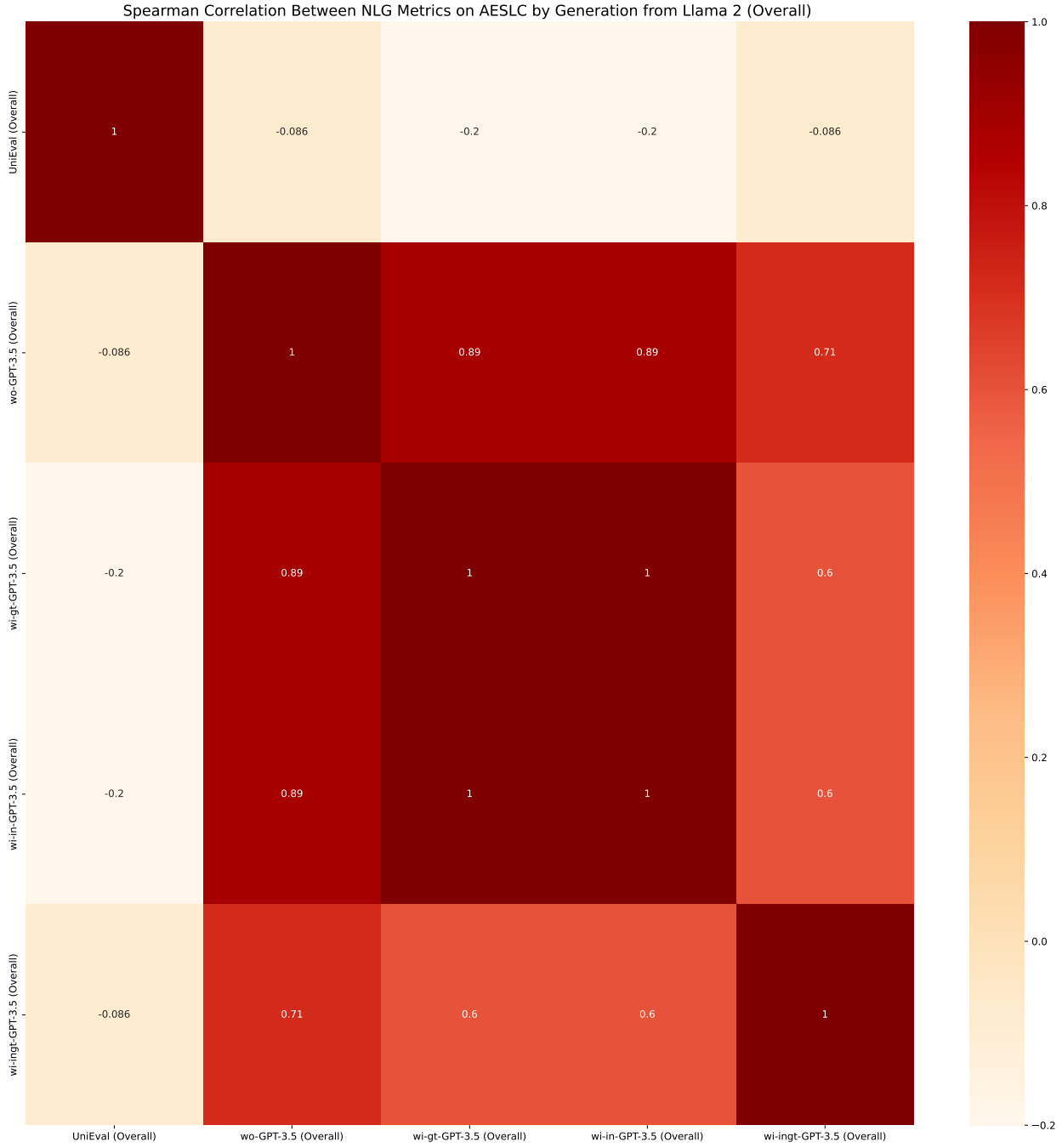

Figure 5.26: Diagram of Spearman correlation in terms of overall between NLG metrics on AESLC dataset from the view of uncertainty estimation methods used in Fig. 5.8. The generated summaries are from Llama 2. For the GPT-3.5-based NLG metrics, we only draw wi-ingt-GPT-3.5 results to save space.

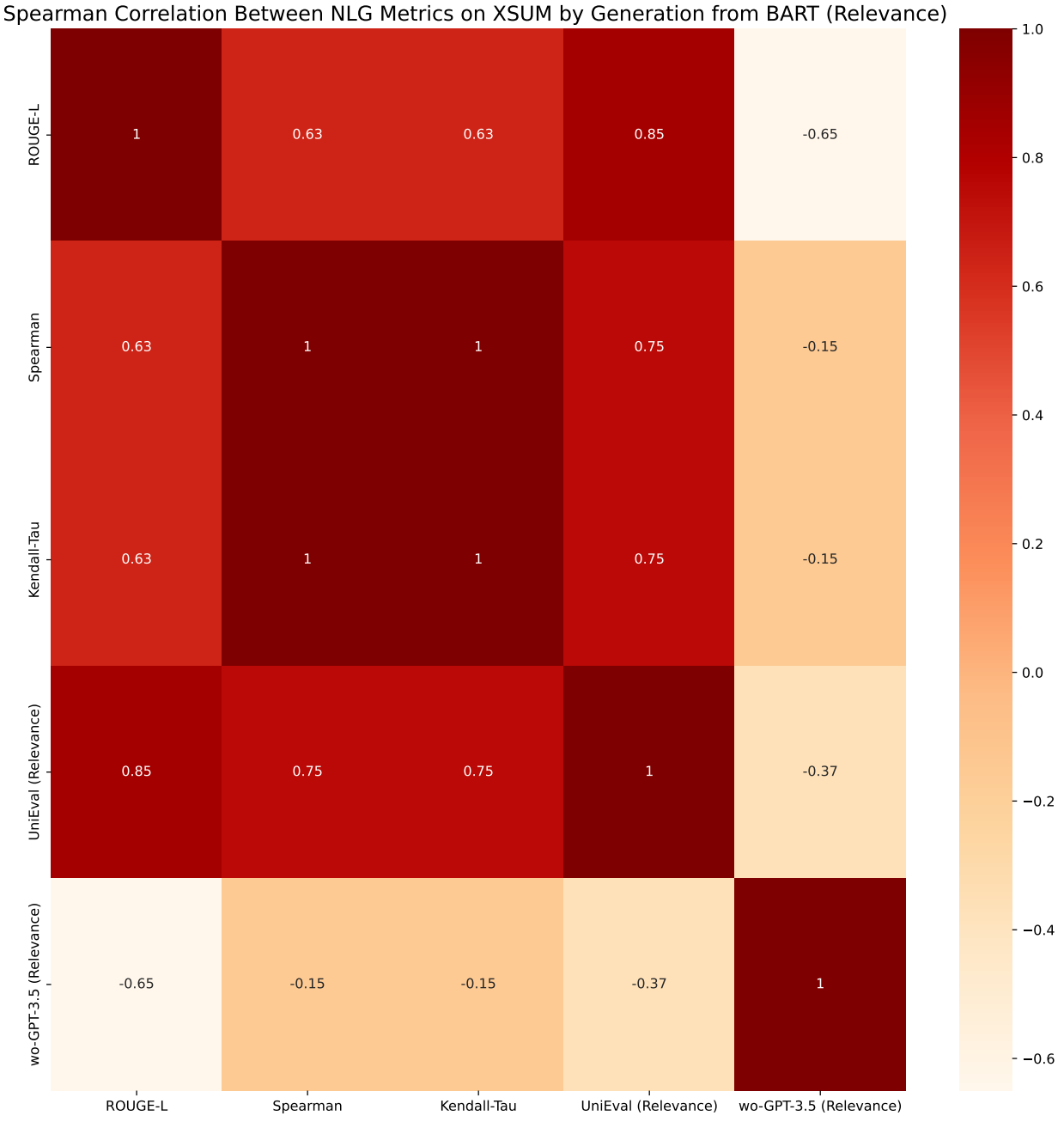

Figure 5.27: Diagram of Spearman correlation in terms of relevance between NLG metrics on XSUM dataset from the view of uncertainty estimation methods used in Fig. 5.12. The generated summaries are from BART. For the GPT-3.5-based NLG metrics, we only conduct wo-GPT-3.5 on the BART generation model setting.

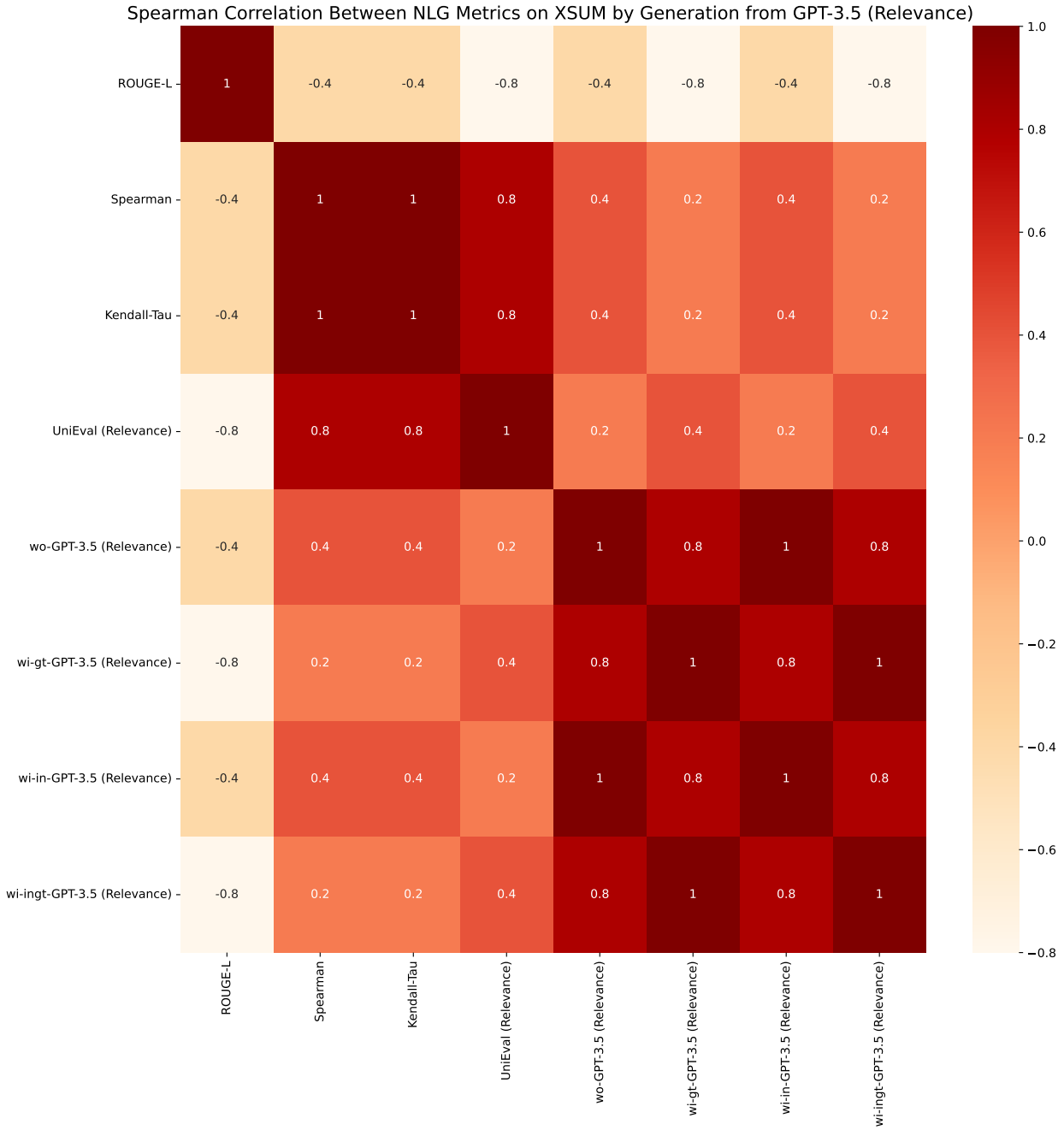

Figure 5.28: Diagram of Spearman correlation in terms of relevance between NLG metrics on XSUM dataset from the view of uncertainty estimation methods used in Fig. 5.13. The generated summaries are from GPT-3.5. For the GPT-3.5-based NLG metrics, we only draw wi-ingt-GPT-3.5 results to save space.

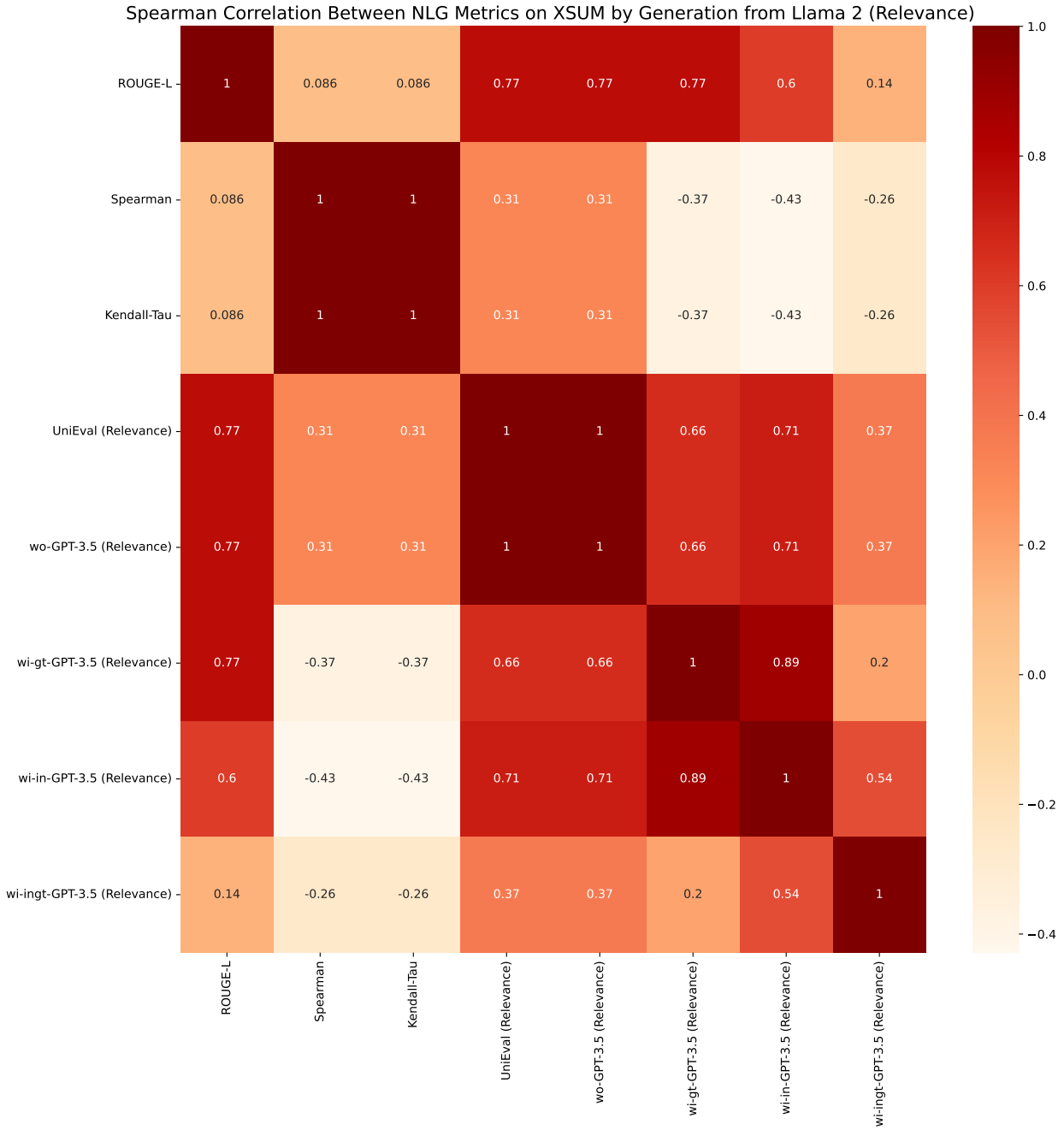

Figure 5.29: Diagram of Spearman correlation in terms of relevance between NLG metrics on XSUM dataset from the view of uncertainty estimation methods used in Fig. 5.14. The generated summaries are from Llama 2. For the GPT-3.5-based NLG metrics, we only draw wi-ingt-GPT-3.5 results to save space.

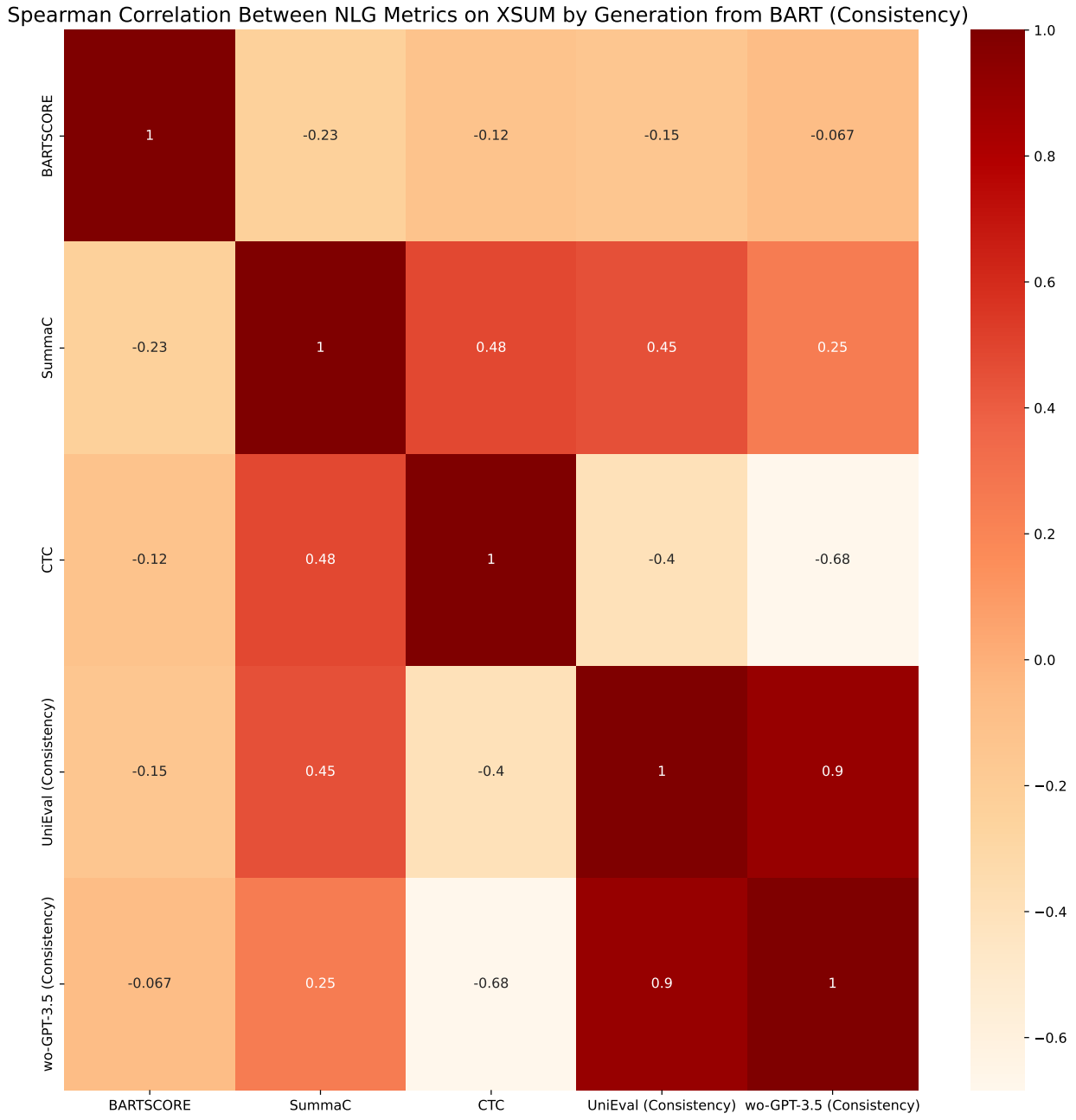

Figure 5.30: Diagram of Spearman correlation in terms of consistency between NLG metrics on XSUM dataset from the view of uncertainty estimation methods used in Fig. 5.12. The generated summaries are from BART. For the GPT-3.5-based NLG metrics, we only conduct wo-GPT-3.5 on the BART generation model setting.

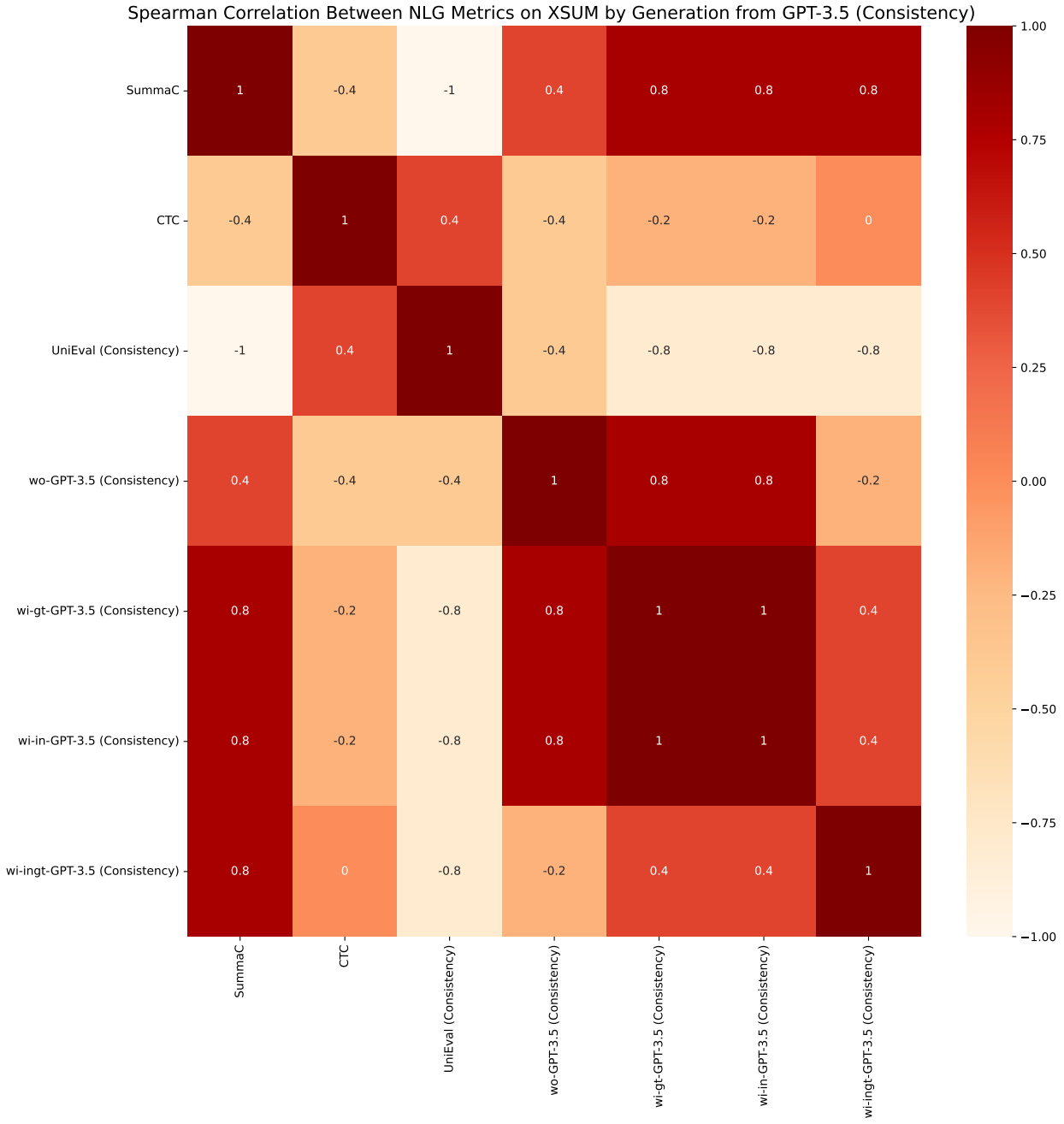

Figure 5.31: Diagram of Spearman correlation in terms of consistency between NLG metrics on XSUM dataset from the view of uncertainty estimation methods used in Fig. 5.13. The generated summaries are from GPT-3.5. For the GPT-3.5-based NLG metrics, we only draw wi-ingt-GPT-3.5 results to save space.

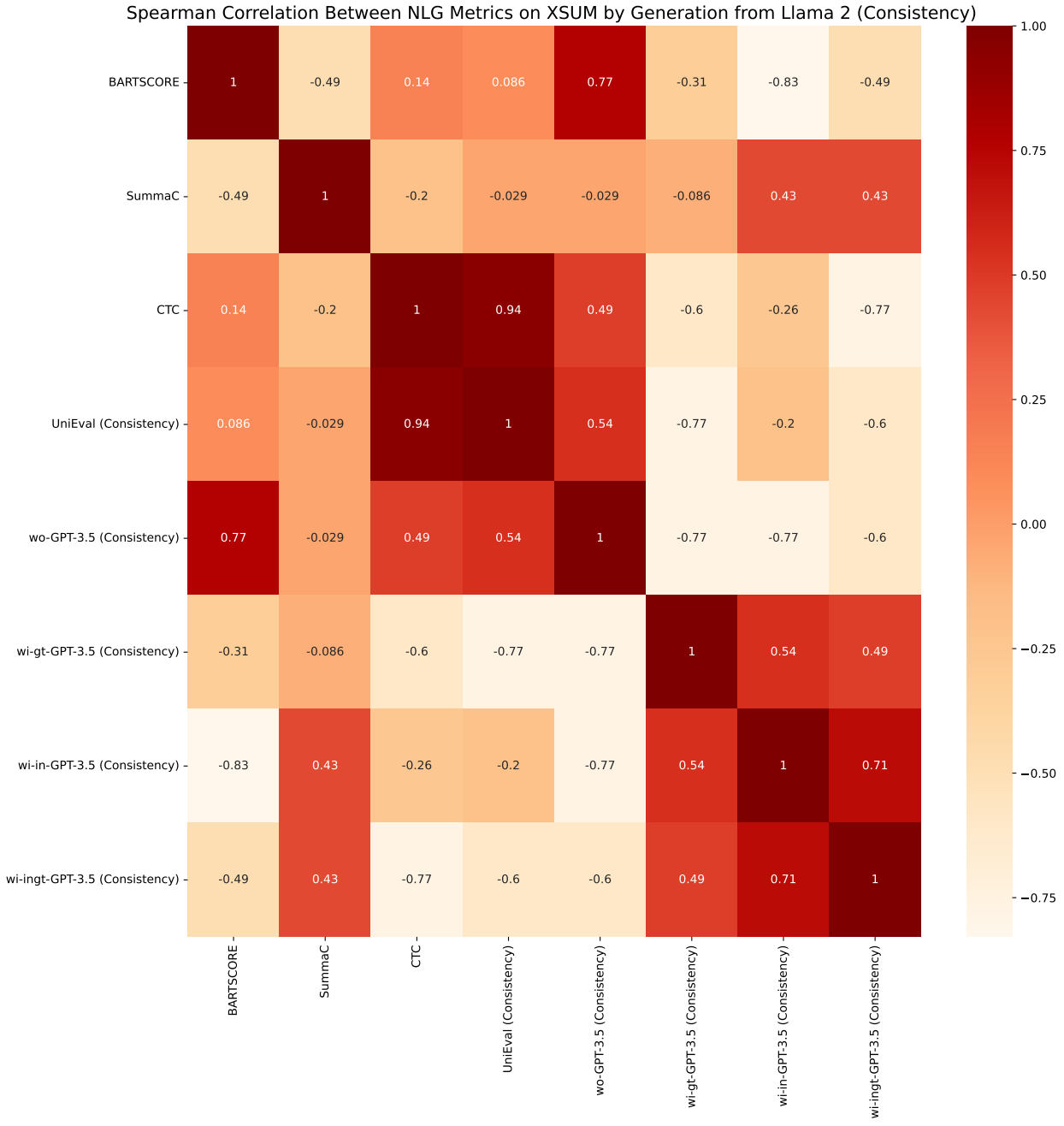

Figure 5.32: Diagram of Spearman correlation in terms of consistency between NLG metrics on XSUM dataset from the view of uncertainty estimation methods used in Fig. 5.14. The generated summaries are from Llama 2. For the GPT-3.5-based NLG metrics, we only draw wi-ingt-GPT-3.5 results to save space.

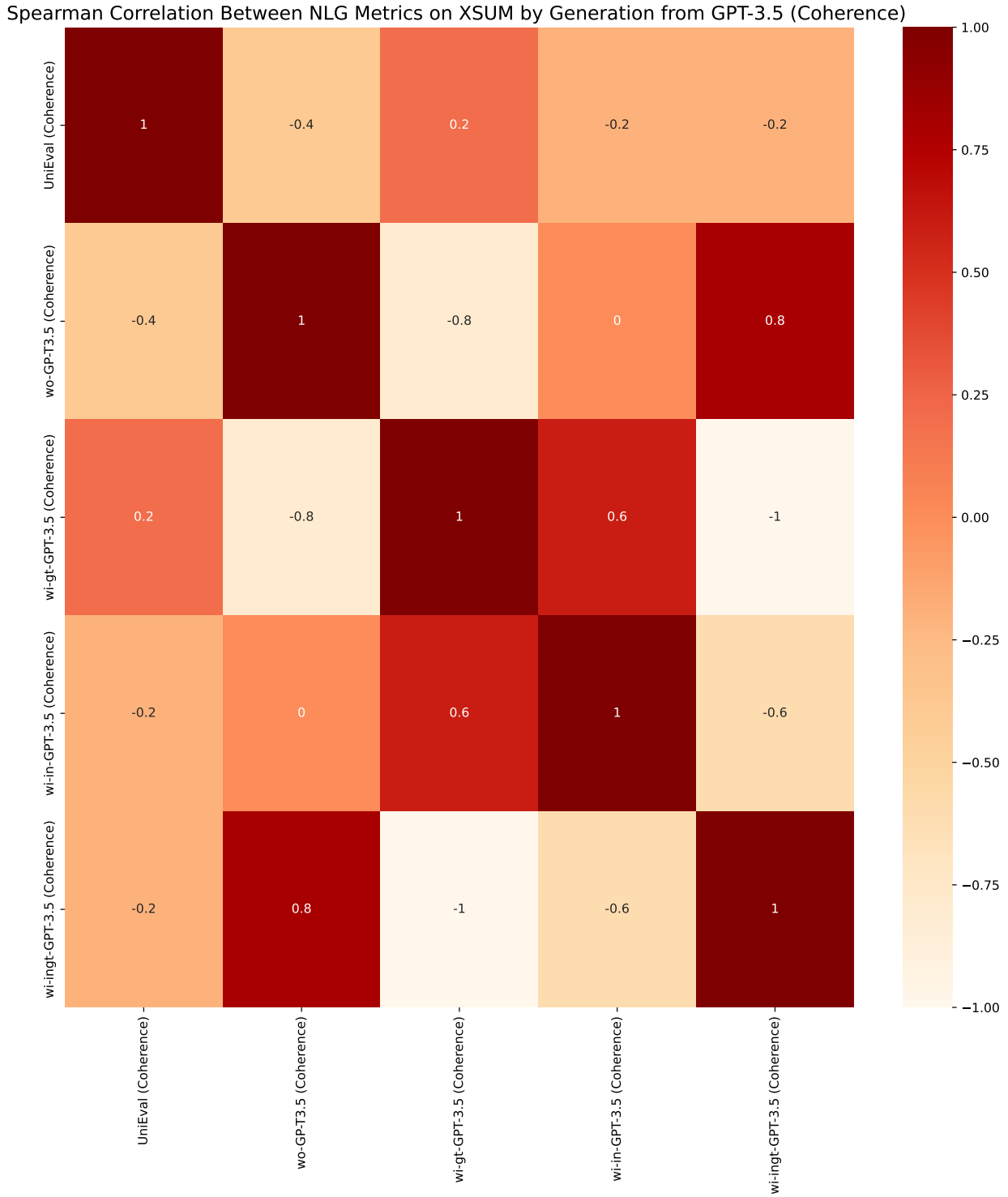

Figure 5.33: Diagram of Spearman correlation in terms of coherence between NLG metrics on XSUM dataset from the view of uncertainty estimation methods used in Fig. 5.13. The generated summaries are from GPT-3.5. For the GPT-3.5-based NLG metrics, we only draw wi-ingt-GPT-3.5 results to save space.

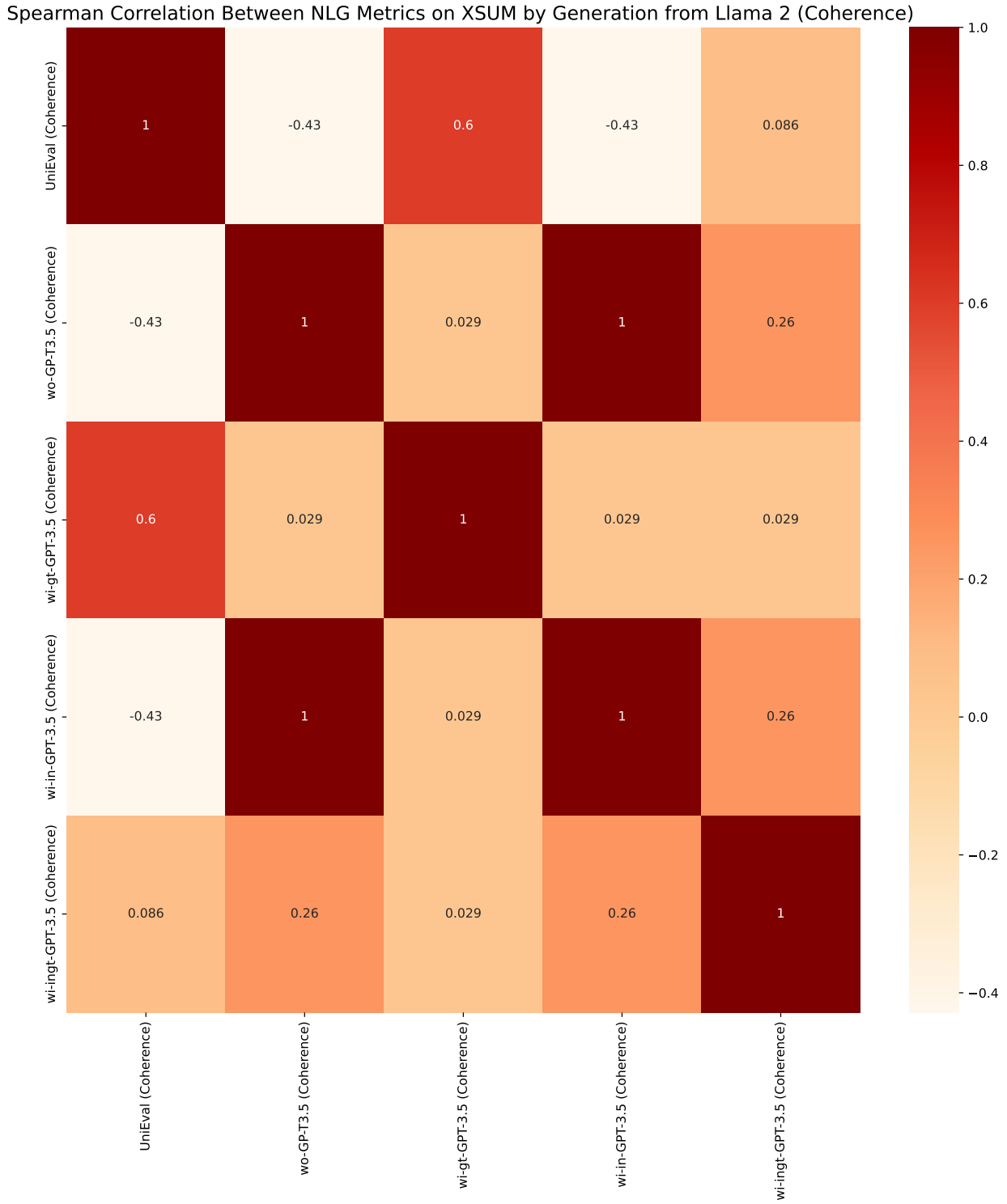

Figure 5.34: Diagram of Spearman correlation in terms of coherence between NLG metrics on XSUM dataset from the view of uncertainty estimation methods used in Fig. 5.14. The generated summaries are from Llama 2. For the GPT-3.5-based NLG metrics, we only draw wi-ingt-GPT-3.5 results to save space.

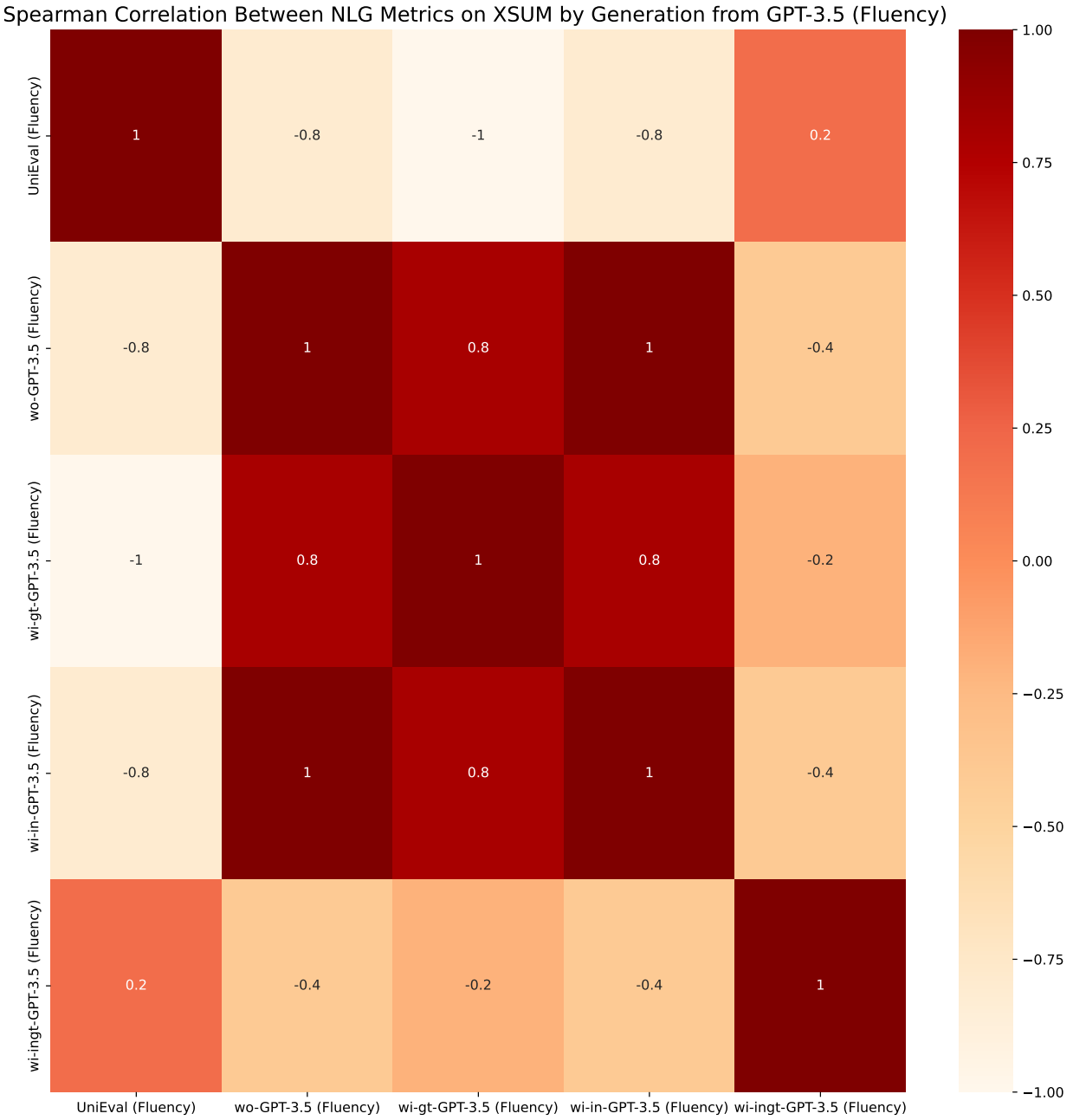

Figure 5.35: Diagram of Spearman correlation in terms of fluency between NLG metrics on XSUM dataset from the view of uncertainty estimation methods used in Fig. 5.13. The generated summaries are from GPT-3.5. For the GPT-3.5-based NLG metrics, we only draw wi-ingt-GPT-3.5 results to save space.

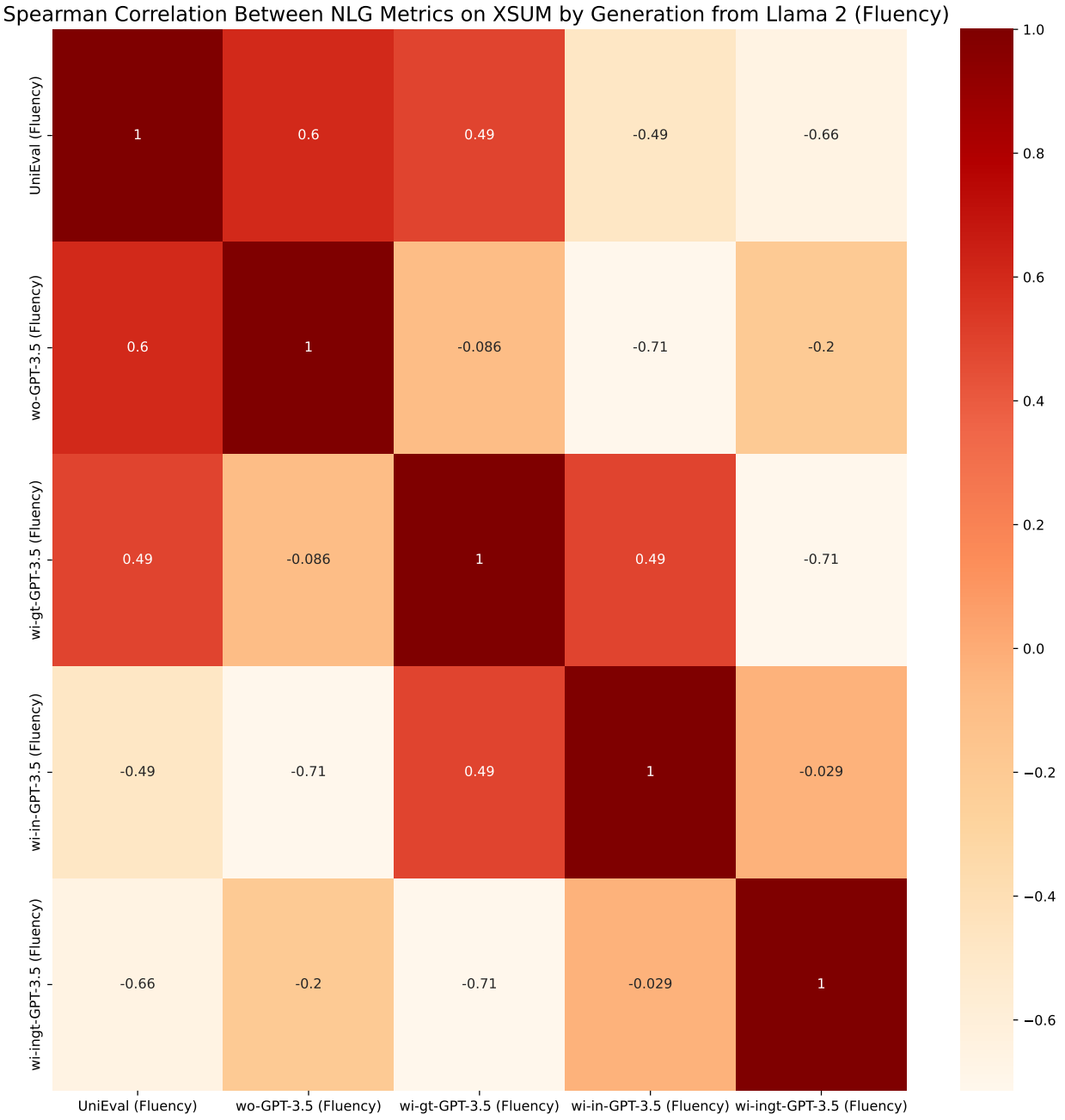

Figure 5.36: Diagram of Spearman correlation in terms of fluency between NLG metrics on XSUM dataset from the view of uncertainty estimation methods used in Fig. 5.14. The generated summaries are from Llama 2. For the GPT-3.5-based NLG metrics, we only draw wi-ingt-GPT-3.5 results to save space.

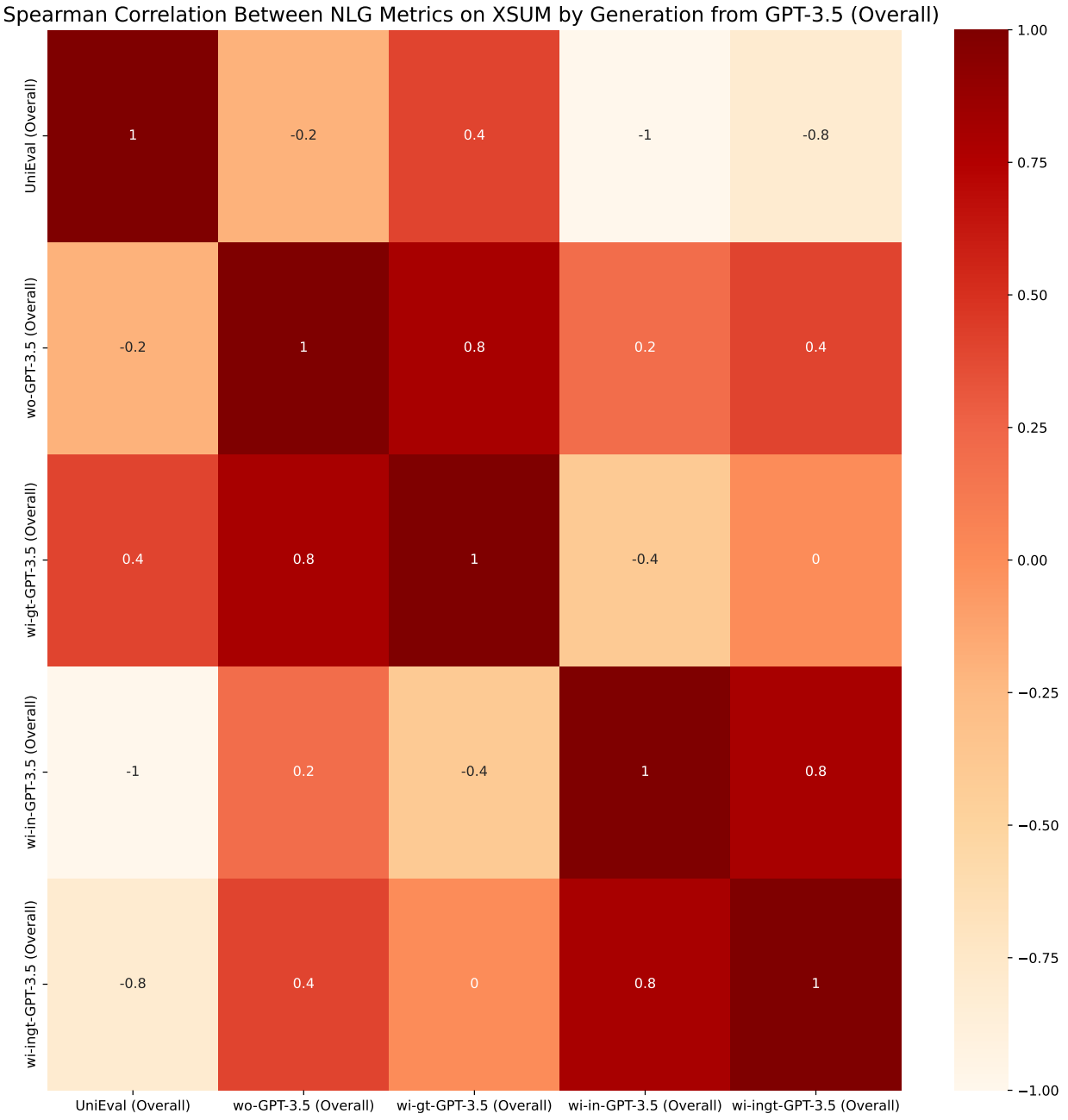

Figure 5.37: Diagram of Spearman correlation in terms of overall between NLG metrics on XSUM dataset from the view of uncertainty estimation methods used in Fig. 5.13. The generated summaries are from GPT-3.5. For the GPT-3.5-based NLG metrics, we only draw wi-ingt-GPT-3.5 results to save space.

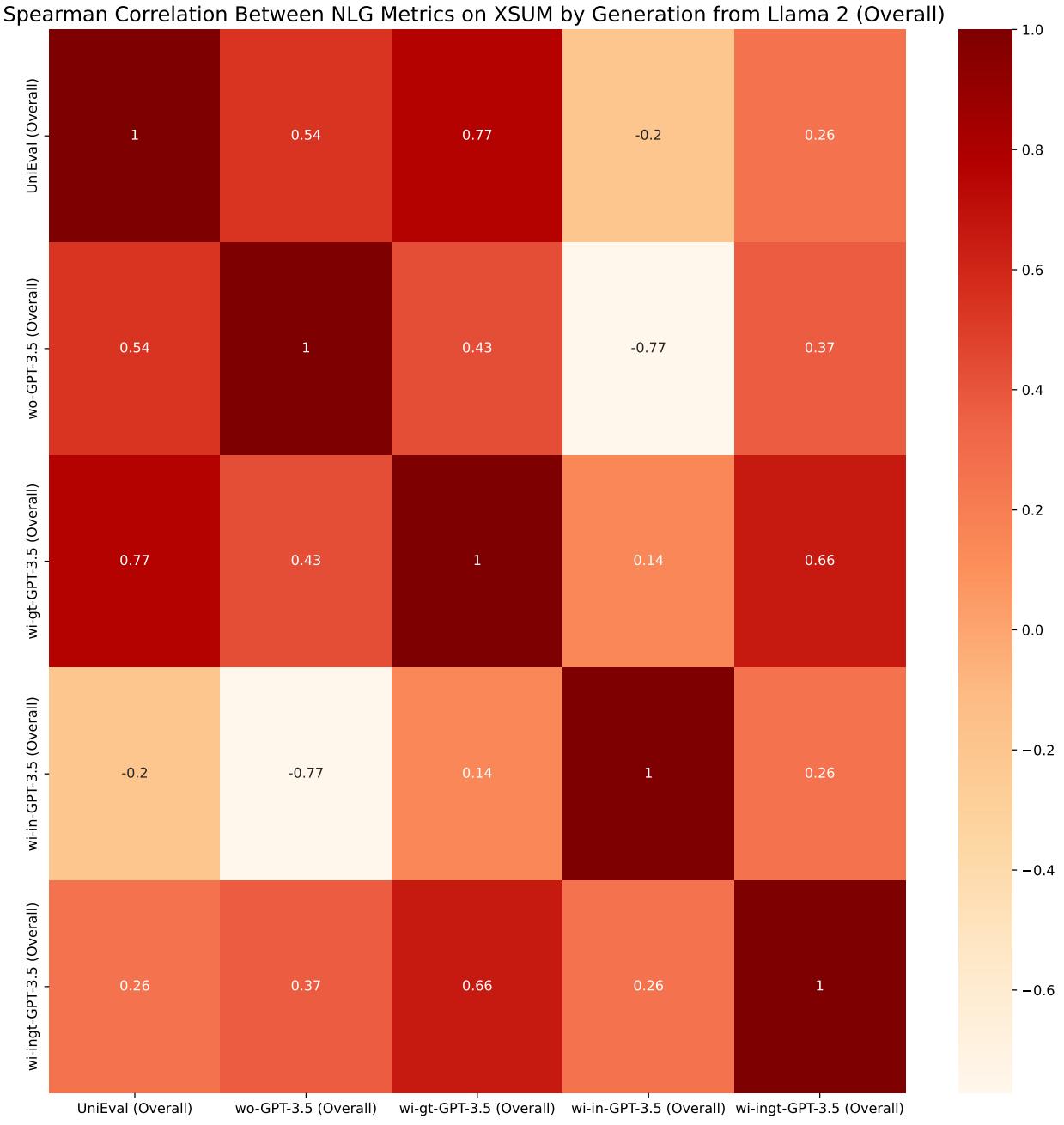

Figure 5.38: Diagram of Spearman correlation in terms of overall between NLG metrics on XSUM dataset from the view of uncertainty estimation methods used in Fig. 5.14. The generated summaries are from Llama 2. For the GPT-3.5-based NLG metrics, we only draw wi-ingt-GPT-3.5 results to save space.

5.6 Summary of Our Findings

Below, we summarize our findings, which are takeaway knowledge. The evidence to obtain these findings is detailed in Sec. 5.4.2.

From the view of **NLG metrics**,

- It is evident that evaluating uncertainty estimation models using different NLG metrics leads to variations in the performance ranking of these models.
- Some evaluations of uncertainty estimation models using different NLG metrics could result in different performance ranks. However, some evaluations of uncertainty estimation models using different NLG metrics may result in the same performance rankings.
- Generation models of the same type across different datasets usually result in similar correlations among various methods. However, this does not apply to the fluency dimension.
- When utilizing LLMs as a type of relevance NLG metric, the choice of target text source can greatly impact the final conclusion. Specifically, using ground-truth summaries versus using input text as the target text source can result in different performance rankings.
- When using LLMs as an NLG metric, if both ground-truth summaries and input text are employed together as the target text source, ground-truth summaries typically impact metric results more than using only the input text in most cases.
- GPT-3.5 knows the concept of relevance, overall. But it might not know the concept of consistency, coherence. As for the concept of fluency, it is hard to tell whether the GPT-3.5 itself understand it or not. Therefore, using GPT-3.5 as an evaluation tool, it would be better to provide the concept definition in the prompt.
- Spearman and Kendall-Tau metrics typically show positive correlations. Therefore, in future experiments, choosing either one of them is sufficient.
- When using LLMs as generation models, UniEval (Relevance) and wo-GPT-3.5 (Relevance) typically show positive correlations with most other NLG metrics. Therefore, either of them could serve as a representative NLG metric in the relevance dimension.
- When faced with scenarios where we must choose between CTC, SummaC, or UniEval, opting for CTC may be preferable due to its positive correlations with the other two in most cases.
- For GPT-3.5-based NLG metrics in the consistency dimension, it is recommended to use wi-in-GPT-3.5 (Consistency) along with one of wo-gt-GPT-3.5, wi-gt-GPT-3.5 (Consistency), or wi-ingt-GPT-3.5 (Consistency).

- Determining the superior GPT-3.5-based metric in the coherence dimension is challenging. Nonetheless, due to their strongly positive correlation, either wi-gt-GPT-3.5 (Coherence) or wi-in-GPT-3.5 (Coherence) could be a suitable choice.
- In the coherence dimension, UniEval (Coherence) could complement either wi-gt-GPT-3.5 (Coherence) or wi-in-GPT-3.5 (Coherence).
- Determining which NLG metrics consistently exhibit positive correlations with others in the fluency dimension is challenging. However, based on their correlation patterns, we recommend wi-gt-GPT-3.5 (Fluency) and wi-ingt-GPT-3.5 (Fluency).
- In the fluency dimension, we also recommend supplementing a GPT-3.5-based NLG metric with UniEval (Fluency).
- In the overall dimension, it is recommended to utilize either wi-in-GPT-3.5 (Overall) or wi-gt-GPT-3.5 (Overall).
- The UniEval (Overall) can serve as an optional supplementary tool for GPT-3.5-based NLG metrics in the overall dimension.

From the view of **uncertainty estimation methods**,

- The future applications of uncertainty estimation methods, focus on one of T-TU, S-TU, and S-RMI could be advantageous, with T-RMI serving as a supplementary measure.
- In future applications of uncertainty estimation methods, when considering MSP and MCSE, opting for one of them and utilizing MTE as an optional supplementary measure could be beneficial.
- In future applications of uncertainty estimation methods, utilizing either MD or RDE alone would suffice rather than using both of them as the baselines.
- In future applications of uncertainty estimation methods involving ECC, LexSim, and EigV, opting for one of them and employing NumSets as a complementary measure could be beneficial.

Chapter 6

Completed Work and Future Work

Natural Language Processing (NLP) has garnered widespread attention. This technology empowers us to perform tasks such as text classification, entity recognition, and even generating responses within a dialogue context. However, it often requires a critical decision: whether to trust a model's predictions. This dissertation is strongly related to the reliability analysis of NLP models. The following sections discuss my accomplishments and future research works in these areas.

6.1 Uncertainty Estimation on (Few-Shot) Text Classification

Text classification is a popular topic with broad applications. A commonly used model for text classification is the Deep Neural Network (DNN). However, some real-world applications require results with higher accuracy than those achieved by state-of-the-art algorithms. Consequently, the most uncertain predictions require domain experts for further decisions [189]. To efficiently utilize limited human resources, it is essential to calculate the *uncertainty score* of the model's prediction, which quantifies the likelihood of false model predictions.

Obtaining the uncertainty score of the model prediction is challenging in three aspects: (1) **How to mitigate the overconfidence of the winning score to enhance its negative correlation with sample uncertainty**. Overconfidence arises because the winning scores of training samples are all set to 1 through one-hot labels, resulting in each sample having the same uncertainty score (e.g., the reciprocal of the winning score) [151]. (2) **How to generate winning scores by considering comprehensive categories of uncertainty at a time**. The process of generating the winning score should account for the impact of different categories of uncertainty simultaneously, as opposed to the previous works [189], which only consider one uncertainty at a time. (3) **How to address uncertainty estima-**

tion on few-shot text classification. The parameter distribution or sample distribution learned in a few-shot setting (e.g., 1 class per class in a 1-shot setting) is inaccurate, rendering distribution-based methods inapplicable to few-shot text classification.

To address the first challenge, we use "mix-up" [151, 183]. Specifically, we employ the mix-up to generate new sample representations with varying winning scores, mitigating over-confidence impact. Plus, we have verified that the uncertainty of the generated sample representation is correlated with the winning scores. To tackle the second challenge, we propose MSD with three components to simultaneously handle epistemic and aleatoric uncertainties [97, 138], thereby enhancing the accuracy of the uncertainty score. Among these components, mix-up is designed to address aleatoric uncertainty. We also introduce 'self-ensembling' to alleviate epistemic uncertainty and 'distinctive score' to quantify epistemic uncertainty. Self-ensembling narrows down feasible parameter sets between models to mitigate parameter uncertainty. The distinctive score calculates the epistemic uncertainty as the distance from a test sample to each training-class sample distribution. To address the third challenge, we propose and utilize uncertainty relations [57] to self-adaptively learn pseudo uncertainty scores, which serve as pseudo ground truth of uncertainty scores. Unlike current contrastive learning models that primarily consider equal relations (i.e., from the same ($=$) or different ($\neq$) classes or instances), our uncertainty relations encompass additional unequal relations, including larger ($>$) or smaller ($<$) uncertainty relations among augmented samples.

In future work, we will focus on two directions. **First, we will handle the interpretability of the uncertainty score**. While the uncertainty score can indicate the reliability of text classification results, we are also keen on identifying which parts of the documents contribute to the uncertainty of a specific model. We plan to explore various methods for selectively removing portions of the document and observing how this affects the predicted uncertainty score. **Second, we will extend our work to Large Language Model (LLM)**, where the training sample distribution is unknown and the distinctive score module is inapplicable. In this case, we want to know whether overconfidence still exists in the LLM and how we can measure the epistemic uncertainty without accessing the training samples in LLM.

6.2 Uncertainty Estimation on Named Entity Recognition

Sequential labeling is a task that involves predicting labels for each token in a sequence, such as Named Entity Recognition (NER). Different from text classification, sequential labeling is a classification task at a more fine-grained level, such as the token or phrase level. We use the NER task as a representative example in sequential labeling. Its objective is to identify spans of phrases and categorize them into specific types (e.g., person, date). NER plays a crucial role in information extraction. However, we still face the question of whether we

should trust the predictions made by the NER model or not. As a result, our focus lies in Uncertainty Estimation on NER.

Two major challenges in uncertainty estimation for NER are as follows: (1) **How to propagate uncertainty from other tokens to the current token.** Unlike uncertainty estimation for text classification, where samples have little to no correlation with other samples, tokens in a sequence are interconnected. Therefore, we need a new framework to estimate token uncertainty within a sequence by considering both the token itself and its connections to other tokens. (2) **How to evaluate uncertainty estimation in different NER prediction scenarios.** Specifically, unlike uncertainty estimation for text classification, where the distinction between false and true predictions is clear-cut, NER tasks involve two components that complicate the definition of false and true predictions. A true NER prediction should first identify a correct span and then assign the correct label to that span.

For the first challenge, we propose a Sequential Labeling Posterior Network (SLPN) that takes into consideration the impact of uncertainty from other tokens. Specifically, inspired by the self-attention mechanism in the Transformer [155], we initially calculate the evidence for each token; subsequently, we accumulate token uncertainty by weighted accumulation of all other token uncertainties in the given text. As a result, uncertainty propagation is achieved. Regarding the second challenge, we have identified the following three cases related to out-of-domain (OOD) detection due to the two components defining true and false predictions in the NER task: (1) the predicted OOD entity exactly matches a true OOD entity, (2) the predicted OOD entity partially matches a true OOD entity in some terms, and (3) the predicted OOD entity does not match a true OOD entity in any terms. For uncertainty estimation in the NER task, we denote the second and third cases as "WS" (Wrong-Spanned). Consequently, uncertainty estimation in NER is evaluated from two perspectives: traditional in-domain and out-of-domain evaluation, as well as wrong span evaluation.

This work initiates a promising research area, and in the future, I will extend the research in the following two directions: **First, we will focus on proposing an evidential neural network that is applicable to Large Language Models (LLMs)**. The current LLM is primarily trained without considering uncertainty and training an LLM with evidential theory remains largely unexplored [41]. **Second, we will explore how to improve both OOD detection performance and wrong span detection performance simultaneously**. The current model exhibits a trade-off between these two tasks. We need to consider additional uncertainty estimation techniques for wrong span detection in the NER task in the future.

6.3 Uncertainty Estimation on Text Summarization

Text summarization generates succinct summaries of dialogues, facilitating users in swiftly grasping key points without delving into intricate contexts [27]. The task of UE-TS has

garnered significant interest among researchers, aiming to gauge the confidence or reliability of the generated summaries [26, 36, 52]. However, an overlooked reliability concern persists concerning the reliability of the UE-TS evaluation framework. This concern arises from two main reasons: Firstly, evaluation metrics for uncertainty estimation in NLG tasks hinge on the alignment between the sample ranks derived from the uncertainty scores and the sample rank based on the respective NLP metric scores. Secondly, the impact of label diversity in NLG tasks, such as text summarization, results in the use of various NLG metrics. Hence, this research focuses on the question, "How does the choice of NLG metric affect the evaluation of uncertainty estimation methods in text summarization?" and its associated questions.

To answer this question and its associate questions, **we construct a comprehensive UE-TS benchmark using two datasets, incorporating fourteen uncertainty estimation methods and twenty-six NLG metrics.** The uncertainty estimation methods encompass **both white-box methods, including four different types of uncertainty estimation methods, and black-box methods**. The NLG metrics cover **four different dimensions of NLG evaluation**. Additionally, our benchmark includes the evaluation of one black-box LLM, one white-box LLM, and one PLM for the text summaries. While it is evident that the current evaluations of UE-TS models lack reliability, **our work contributes to a deeper understanding of the relationship between NLG metrics and uncertainty estimation methods, particularly in terms of their effectiveness and interplay.** This understanding can guide future research and inform the efficient development of more effective uncertainty estimation methods and evaluation protocols for UE-TS or other NLG tasks.

Future directions of my research on uncertainty estimation in text summarization encompass three main aspects. **First, we aim to design uncertainty estimation metrics that are independent of NLG metrics or develop metrics that closely align with human judgment criteria.** Our observation is that the current evaluation of UE-TS models varies depending on the chosen NLG metrics underscores. Thus, there is a need for robust uncertainty estimation metrics that can withstand such variations or even align more closely with human judgment. **Second, we are intrigued by exploring uncertainty estimation for multi-round dialogue responses, extending beyond our current focus on single-round generation, such as text summarization.** Understanding uncertainty transmission in multi-turn dialogues is pivotal for advancing dialogue systems. **Furthermore, we are eager to delve into uncertainty estimation for multimodal dialogues.** In domains like e-commerce, where images complement text, addressing uncertainty across modalities poses a significant challenge. This entails researching effective techniques to integrate or convey uncertainty across different modalities.

6.4 Previous Papers Before Virginia Tech

1. (Neurocomputing 2018 [50]) **J. He**, B. Ma, S. Wang, Y. Liu and Q. Huang. Multi-label Double Layers Learning for Cross-Modal Retrieval . Neurocomputing, 2018, 275: 1893-1902.

2. (Workshop on MASS 2017 [53]) **J. He**, S. Wang, Q. Qu, W. Zhang and Q. Huang. Efficient Cross-Modal Retrieval Using Social Tag Information Towards Mobile Applications . International Workshop on Mobility Analytics for Spatio-temporal and Social Data. Springer, Cham, 2017: 157-176.

3. (IJCAI 2017 [184]) L. Zhang, B. Ma, **J. He**, G. Li, Q. Huang and Q. Tian. Adaptively Unified Semi-supervised Learning for Cross-Modal Retrieval . Twenty-Sixth International Joint Conference on Artificial Intelligence, 2017: 3406-3412.

4. (ACM Multimedia 2016 [49]) **J. He**, B. Ma, S. Wang, Y. Liu and Q. Huang. Cross-modal Retrieval by Real Label Partial Least Squares . Proceedings of the 2016 ACM on Multimedia Conference: 227-231.

6.5 Current Papers At Virginia Tech

Published Journal Papers

1. (Neurocomputing 2022 [59]) **J. He**, X. Zhang, S. Lei, S. Wang, Q. Huang, C. Lu, B. Xiao. Semantic inpainting on segmentation map via multi-expansion loss. Neurocomputing 501 (2022): 306-317.

2. (Frontiers in Environmental Science [12]) F. Chen, D. Wang, S. Lei, **J. He**, Y. Fu, C. Lu. "Adaptive graph convolutional imputation network for environmental sensor data recovery."Frontiers in Environmental Science 10 (2022): 2120.

Published Conference Papers

3. (NAACL 2024 [52]) **J. He**, H. Su, J. Cai, I. Shalyminov, H. Song, S. Mansour. Semi-Supervised Dialogue Abstractive Summarization via High-Quality Pseudolabel Selection. Accepted by NAACL 2024 main conference.

4. (NAACL Findings 2024 [56]) **J. He**, L. Yu, S. Lei, C. Lu, F. Chen. Uncertainty Estimation on Sequential Labeling via Uncertainty Transmission. Accepted by NAACL 2024 Findings.

5. (EACL 2024 [51]) **J. He**, J. Salazar, K. Yao, H. Li, J. Cai. Zero-Shot End-to-End Spoken Language Understanding via Cross-Modal Selective Self-Training. Accepted by EACL 2024 main conference.

6. (KDD 2023 [57]) **J. He**, X. Zhang, S. Lei, F. Chen, A. Alhamadani, B. Xiao, C. Lu. CLUR: Uncertainty Estimation for Few-Shot Text Classification with Contrastive Learning. In Proceedings of the 29th ACM SIGKDD international conference on knowledge discovery & data mining.

7. (SIGIR 2023, demo [54]) **J. He**, S. Wu, A. Alhamadani, C. Chen, W. Lu, C. Lu, D. Solnick, Y. Li. Metro-Scope: An Advanced System for Real-Time Detection and Analysis of Metro-Related Threats and Events via Twitter. In Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval: 3130-3134.

8. (ICCV workshop 2021 [55]) **J. He**, B. Xiao, X. Zhang, S. Wang, S. Lei, and C. Lu. Reducing noise pixels and metric bias in semantic inpainting on segmentation map. In Proceedings of the IEEE/CVF International Conference on Computer Vision 2021 (pp. 1876-1885).

9. (EMNLP 2020 [58]) **J. He**, X. Zhang, S. Lei, Z. Chen, F. Chen, A. Alhamadani, B. Xiao and C. Lu. Towards More Accurate Uncertainty Estimation In Text Classification . In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: 8362-8372.

10. (ICME 2021 [86]) S. Lei, X. Zhang, **J. He**, F. Chen, C. Lu. Few-Shot Semantic Segmentation via Prototype Augmentation with Image-Level Annotations. In 2021 IEEE International Conference on Multimedia and Expo. 2021 Jul 5 (pp. 1-6). IEEE.

11. (NAACL Findings 2022 [87]) S. Lei, X. Zhang, **J. He**, F. Chen, C. Lu. Uncertainty-Aware Cross-Lingual Transfer with Pseudo Partial Labels. In Findings of the Association for Computational Linguistics: NAACL 2022 (pp. 1987-1997).

12. (ECCV 2022 [85]) S. Lei, X. Zhang, **J. He**, F. Chen, C. Lu. Cross-Domain Few-Shot Semantic Segmentation. Cross-Domain Few-Shot Semantic Segmentation. In Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, pp. 73-90.

13. (ACL 2023 [88]) S. Lei, X. Zhang, **J. He**, F. Chen, C. Lu. TART: Improved Few-shot Text Classification Using Task-Adaptive Reference Transformation. In 61st Annual Meeting of the Association for Computational Linguistics.

14. (SIGSPATIAL 2023 [160]) L. Wang, S. Lei, **J. He**, S. Wang, M. Zhang, C. Lu. Self-Correlation and Cross-Correlation Learning for Few-Shot Remote Sensing Image Semantic Segmentation.

15. (ASONAM 2023, Industry Track [2]) A. Abdulaziz, K. Althubiti, S. Sharker, **J. He**, L. Alkulaib, S. Behal, M. Khan, C. Lu. From Guest to Family: An Innovative Framework for Enhancing Memorable Experiences in the Hotel Industry.

16. (ArabicNLP 2023 [3]) A. Abdulaziz, X. Zhang, **J. He**, C. Lu. LANS: Large-scale Arabic News Summarization Corpus. ArabicNLP 2023.

17. (NAACL 2024, Industry Track) S. Wang, T. Ji, **J. He**, M. Almutairi, D. Wang, L. Wang, M. Zhang, C. Lu. AMA-LSTM: Pioneering Robust and Fair Financial Audio Analysis for Stock Volatility Prediction. Accepted by NAACL Industry Track 2024.

18. (ICASSP 2024 [186]) M. Zhang, **J. He**, S. Lei, M. Yue, L. Wang, C. Lu. Can LLM Find The Green Circle? Investigation And Human-Guided Tool Manipulation For Compositional Generalization. ICASSP 2024.

19. (ISGT 2024 [193]) Z Zhang, **J. He**, A. Kumar, S. Rahman. AI-based Space Occupancy Estimation Using Environmental Sensor Data. 2024 IEEE Power & Energy Society Innovative Smart Grid Technologies Conference (ISGT). IEEE, 2024.

ArXived Works

20. [149] Y. Sun (co-first author), **J. He** (co-first author), L. Cui, S. Lei, C. Lu. Med-MMHL: A Multi-Modal Dataset for Detecting Human- and LLM-Generated Misinformation in the Medical Domain. ArXiv preprint arXiv:2306.08871 (2023).

21. [148] Y. Sun, **J. He**, L. Cui, S. Lei, C. Lu. Exploring the Deceptive Power of LLM-Generated Fake News: A Study of Real-World Detection Challenges. ArXiv preprint arXiv:2403.18249 (2024).

22. [185] M. Zhang, **J. He**, T. Ji, C. Lu. Don't Go To Extremes: Revealing the Excessive Sensitivity and Calibration Limitations of LLMs in Implicit Hate Speech Detection. ArXiv preprint arXiv:2402.11406 (2024).

Planned Submission

23. (Chapter 5) **J. He**, S. Lei, X. Zhang, B. Xiao. C. Lu, Can We Trust the Performance Evaluation of Uncertainty Estimation Methods in Text Summarization? (Planned for NIPS 2024).

24. **J. He**, L. Yu, S. Lei, M. Zhang, F. Cheng, K. Ding, C. Lu. A survey for the Uncertainty Estimation in NLU and NLG under LLM settings. (Planned for ACM Survery).

Bibliography

[1] Akbik, A., Bergmann, T., Blythe, D., Rasul, K., Schweter, S., Vollgraf, R.: FLAIR: An easy-to-use framework for state-of-the-art NLP. In: NAACL 2019, 2019 Annual Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations), pp. 54–59 (2019)

[2] Alhamadani, A., Althubiti, K., Sarkar, S., He, J., Alkulaib, L., Behal, S., Khan, M., Lu, C.T.: From Guest to Family: An Innovative Framework for Enhancing Memorable Experiences in the Hotel Industry. In: Proceedings of the International Conference on Advances in Social Networks Analysis and Mining, pp. 492–501 (2023)

[3] Alhamadani, A., Zhang, X., He, J., Lu, C.T.: LANS: Large-scale Arabic News Summarization Corpus. arXiv preprint arXiv:2210.13600 (2022)

[4] Amini, A., Schwarting, W., Soleimany, A., Rus, D.: Deep evidential regression. Advances in Neural Information Processing Systems **33**, 14927–14937 (2020)

[5] Antorán, J., Allingham, J.U., Hernández-Lobato, J.M.: Depth uncertainty in neural networks. arXiv preprint arXiv:2006.08437 (2020)

[6] Bao, Y., Wu, M., Chang, S., Barzilay, R.: Few-shot Text Classification with Distributional Signatures. In: International Conference on Learning Representations (2020)

[7] Caron, M., Misra, I., Mairal, J., Goyal, P., Bojanowski, P., Joulin, A.: Unsupervised learning of visual features by contrasting cluster assignments. arXiv preprint arXiv:2006.09882 (2020)

[8] Chang, H.S., Vembu, S., Mohan, S., Uppaal, R., McCallum, A.: Using error decay prediction to overcome practical issues of deep active learning for named entity recognition. Machine Learning **109**, 1749–1778 (2020)

[9] Charpentier, B., Borchert, O., Zügner, D., Geisler, S., Günnemann, S.: Natural Posterior Network: Deep Bayesian Uncertainty for Exponential Family Distributions. ICRL (2022)

[10] Charpentier, B., Zügner, D., Günnemann, S.: Posterior network: Uncertainty estimation without ood samples via density-based pseudo-counts. Advances in Neural Information Processing Systems **33**, 1356–1367 (2020)

[11] Chen, B., Jiang, J., Wang, X., Wang, J., Long, M.: Debiased pseudo labeling in self-training. arXiv preprint arXiv:2202.07136 (2022)

[12] Chen, F., Wang, D., Lei, S., He, J., Fu, Y., Lu, C.T.: Adaptive graph convolutional imputation network for environmental sensor data recovery. Frontiers in Environmental Science **10**, 1025268 (2022)

[13] Chen, T., Guestrin, C.: Xgboost: A scalable tree boosting system. In: Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining, pp. 785–794 (2016)

[14] Chen, T., Kornblith, S., Norouzi, M., Hinton, G.: A simple framework for contrastive learning of visual representations. In: International conference on machine learning, pp. 1597–1607. PMLR (2020)

[15] Chen, X., Fan, H., Girshick, R., He, K.: Improved baselines with momentum contrastive learning. arXiv preprint arXiv:2003.04297 (2020)

[16] Chen, X., He, K.: Exploring simple siamese representation learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 15750–15758 (2021)

[17] Chiu, J.P., Nichols, E.: Named entity recognition with bidirectional LSTM-CNNs. Transactions of the association for computational linguistics **4**, 357–370 (2016)

[18] Chuang, Y.N., Tang, R., Jiang, X., Hu, X.: SPeC: A soft prompt-based calibration on performance variability of large language model in clinical notes summarization. Journal of Biomedical Informatics p. 104606 (2024)

[19] davidberenstein1957: A public medical domain dataset (2023). URL `https://huggingface.co/datasets/argilla/medical-domain`

[20] Deng, M., Tan, B., Liu, Z., Xing, E.P., Hu, Z.: Compression, transduction, and creation: A unified framework for evaluating natural language generation. arXiv preprint arXiv:2109.06379 (2021)

[21] Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805 (2018)

[22] Dong, B., Wang, Y., Sun, H., Wang, Y., Hashemi, A., Du, Z.: CML: A contrastive meta learning method to estimate human label confidence scores and reduce data collection cost. In: Proceedings of The Fifth Workshop on e-Commerce and NLP (ECNLP 5), pp. 35–43 (2022)

[23] Dong, B., Wu, Y., Yeh, M., Lin, Y., Chen, Y., Yang, H., Wang, F., Bai, W., Brahmkstri, K., Yimin, Z., et al.: Semi-supervised Context Discovery for Peer-Based Anomaly Detection in Multi-layer Networks. In: Information and Communications Security: 24th International Conference, ICICS 2022, Canterbury, UK, September 5–8, 2022, Proceedings, pp. 508–524. Springer (2022)

[24] Dong, L., Quirk, C., Lapata, M.: Confidence modeling for neural semantic parsing. arXiv preprint arXiv:1805.04604 (2018)

[25] Ebrahimi, J., Rao, A., Lowd, D., Dou, D.: Hotflip: White-box adversarial examples for text classification. arXiv preprint arXiv:1712.06751 (2017)

[26] Fadeeva, E., Vashurin, R., Tsvigun, A., Vazhentsev, A., Petrakov, S., Fedyanin, K., Vasilev, D., Goncharova, E., Panchenko, A., Panov, M., et al.: LM-Polygraph: Uncertainty Estimation for Language Models. In: Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, pp. 446–461 (2023)

[27] Feng, X., Feng, X., Qin, B.: A survey on dialogue summarization: Recent advances and new frontiers. arXiv preprint arXiv:2107.03175 (2021)

[28] Floridi, L., Chiriatti, M.: GPT-3: Its nature, scope, limits, and consequences. Minds and Machines **30**, 681–694 (2020)

[29] Fomicheva, M., Sun, S., Yankovskaya, L., Blain, F., Guzmán, F., Fishel, M., Aletras, N., Chaudhary, V., Specia, L.: Unsupervised quality estimation for neural machine translation. Transactions of the Association for Computational Linguistics **8**, 539–555 (2020)

[30] Gal, Y., Ghahramani, Z.: Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In: international conference on machine learning, pp. 1050–1059 (2016)

[31] Gales, M., Malinin, A.: UNCERTAINTY ESTIMATION IN AUTOREGRESSIVE STRUCTURED PREDICTION (2021)

[32] Ge, Y., Guo, Y., Yang, Y.C., Al-Garadi, M.A., Sarker, A.: Few-shot learning for medical text: A systematic review. arXiv preprint arXiv:2204.14081 (2022)

[33] Geng, R., Li, B., Li, Y., Sun, J., Zhu, X.: Dynamic memory induction networks for few-shot text classification. arXiv preprint arXiv:2005.05727 (2020)

[34] Geng, R., Li, B., Li, Y., Zhu, X., Jian, P., Sun, J.: Induction networks for few-shot text classification. arXiv preprint arXiv:1902.10482 (2019)

[35] Gidiotis, A., Tsoumakas, G.: Bayesian active summarization. arXiv preprint arXiv:2110.04480 (2021)

[36] Gidiotis, A., Tsoumakas, G.: Should we trust this summary? Bayesian abstractive summarization to the rescue. arXiv preprint arXiv:2105.10155 (2021)

[37] Gómez-Guillamón, A.D.: The usefulness of the audit report in investment and financing decisions. Managerial auditing journal **18**(6/7), 549–559 (2003)

[38] Grill, J.B., Strub, F., Altché, F., Tallec, C., Richemond, P.H., Buchatskaya, E., Doersch, C., Pires, B.A., Guo, Z.D., Azar, M.G., et al.: Bootstrap your own latent: A new approach to self-supervised learning. arXiv preprint arXiv:2006.07733 (2020)

[39] Gunel, B., Du, J., Conneau, A., Stoyanov, V.: Supervised contrastive learning for pre-trained language model fine-tuning. International Conference on Learning Representations (2021)

[40] Guo, C., Pleiss, G., Sun, Y., Weinberger, K.Q.: On calibration of modern neural networks. In: Proceedings of the 34th International Conference on Machine Learning-Volume 70, pp. 1321–1330. JMLR. org (2017)

[41] Guo, Z., Wan, Z., Zhang, Q., Zhao, X., Zhang, Q., Kaplan, L.M., Jøsang, A., Jeong, D.H., Chen, F., Cho, J.H.: A survey on uncertainty reasoning and quantification in belief theory and its application to deep learning. Information Fusion p. 101987 (2023)

[42] Gupta, A., Thadani, K., O'Hare, N.: Effective few-shot classification with transfer learning. In: Proceedings of the 28th International Conference on Computational Linguistics, pp. 1061–1066 (2020)

[43] Gupta, V., Lehal, G.S.: A survey of text summarization extractive techniques. Journal of emerging technologies in web intelligence **2**(3), 258–268 (2010)

[44] Hama, K., Matsubara, T., Uehara, K., Cai, J.: Exploring uncertainty measures for image-caption embedding-and-retrieval task. arXiv preprint arXiv:1904.08504 (2019)

[45] Hamisu, M., Mansour, A.: Detecting advance fee fraud using nlp bag of word model. In: 2020 IEEE 2nd International Conference on Cyberspac (CYBER NIGERIA), pp. 94–97. IEEE (2021)

[46] Hammerton, J.: Named entity recognition with long short-term memory. In: Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003, pp. 172–175 (2003)

[47] Han, C., Fan, Z., Zhang, D., Qiu, M., Gao, M., Zhou, A.: Meta-learning adversarial domain adaptation network for few-shot text classification. arXiv preprint arXiv:2107.12262 (2021)

[48] Hart, R.A., Yu, L., Lou, Y., Chen, F.: Improvements on Uncertainty Quantification for Node Classification via Distance-Based Regularization (2023)

[49] He, J., Ma, B., Wang, S., Liu, Y., Huang, Q.: Cross-modal retrieval by real label partial least squares. In: Proceedings of the 24th ACM international conference on Multimedia, pp. 227–231 (2016)

[50] He, J., Ma, B., Wang, S., Liu, Y., Huang, Q.: Multi-label double-layer learning for cross-modal retrieval. Neurocomputing **275**, 1893–1902 (2018)

[51] He, J., Salazar, J., Yao, K., Li, H., Cai, J.: Zero-Shot End-to-End Spoken Language Understanding via Cross-Modal Selective Self-Training. arXiv preprint arXiv:2305.12793 (2023)

[52] He, J., Su, H., Cai, J., Shalyminov, I., Song, H., Mansour, S.: Semi-Supervised Dialogue Abstractive Summarization via High-Quality Pseudolabel Selection. arXiv preprint arXiv:2403.04073 (2024)

[53] He, J., Wang, S., Qu, Q., Zhang, W., Huang, Q.: Efficient Cross-Modal Retrieval Using Social Tag Information Towards Mobile Applications. In: Mobility Analytics for Spatio-Temporal and Social Data: First International Workshop, MATES 2017, Munich, Germany, September 1, 2017, Revised Selected Papers 1, pp. 157–176. Springer (2018)

[54] He, J., Wu, S.Y., Alhamadani, A., Chen, C.F., Lu, W.F., Lu, C.T., Solnick, D., Li, Y.: MetroScope: An Advanced System for Real-Time Detection and Analysis of Metro-Related Threats and Events via Twitter. In: Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 3130–3134 (2023)

[55] He, J., Xiao, B., Zhang, X., Lei, S., Wang, S., Lu, C.T.: Reducing noise pixels and metric bias in semantic inpainting on segmentation map. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 1876–1885 (2021)

[56] He, J., Yu, L., Lei, S., Lu, C.T., Chen, F.: Uncertainty Estimation on Sequential Labeling via Uncertainty Transmission. arXiv preprint arXiv:2311.08726 (2023)

[57] He, J., Zhang, X., Lei, S., Alhamadani, A., Chen, F., Xiao, B., Lu, C.T.: CLUR: Uncertainty Estimation for Few-Shot Text Classification with Contrastive Learning. In: Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, pp. 698–710 (2023)

[58] He, J., Zhang, X., Lei, S., Chen, Z., Chen, F., Alhamadani, A., Xiao, B., Lu, C.: Towards More Accurate Uncertainty Estimation In Text Classification. In: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 8362–8372 (2020)

[59] He, J., Zhang, X., Lei, S., Wang, S., Lu, C.T., Xiao, B.: Semantic inpainting on segmentation map via multi-expansion loss. Neurocomputing **501**, 306–317 (2022)

[60] He, R., McAuley, J.: Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In: proceedings of the 25th international conference on world wide web, pp. 507–517 (2016)

[61] Hendrycks, D., Gimpel, K.: A baseline for detecting misclassified and out-of-distribution examples in neural networks. arXiv preprint arXiv:1610.02136 (2016)

[62] Hong, S., Jang, T.Y.: LEA: Meta Knowledge-Driven Self-Attentive Document Embedding for Few-Shot Text Classification. In: Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 99–106 (2022)

[63] Hu, Y., Khan, L.: Uncertainty-Aware Reliable Text Classification. In: Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, pp. 628–636 (2021)

[64] Hu, Y., Ou, Y., Zhao, X., Cho, J.H., Chen, F.: Multidimensional Uncertainty-Aware Evidential Neural Networks. arXiv preprint arXiv:2012.13676 (2020)

[65] Huang, Z., Xu, W., Yu, K.: Bidirectional LSTM-CRF models for sequence tagging. arXiv preprint arXiv:1508.01991 (2015)

[66] Jabreel, M., Hassan, F., Moreno, A.: Target-dependent sentiment analysis of tweets using bidirectional gated recurrent neural networks. In: Advances in Hybridization of Intelligent Methods, pp. 39–55. Springer (2018)

[67] Jagannatha, A., Yu, H.: Calibrating Structured Output Predictors for Natural Language Processing. arXiv preprint arXiv:2004.04361 (2020)

[68] Jiang, W., Zhao, Y., Wu, Y., Zuo, H.: Capturing Model Uncertainty with Data Augmentation in Deep Learning. In: Proceedings of the 2022 SIAM International Conference on Data Mining (SDM), pp. 271–279. SIAM (2022)

[69] Jiang, X., Havaei, M., Chartrand, G., Chouaib, H., Vincent, T., Jesson, A., Chapados, N., Matwin, S.: On the importance of attention in meta-learning for few-shot text classification. arXiv preprint arXiv:1806.00852 (2018)

[70] Joulin, A., Grave, E., Bojanowski, P., Douze, M., Jégou, H., Mikolov, T.: Fasttext. zip: Compressing text classification models. arXiv preprint arXiv:1612.03651 (2016)

[71] Jsang, A.: Subjective Logic: A formalism for reasoning under uncertainty. Springer Publishing Company, Incorporated (2018)

[72] Kadavath, S., Conerly, T., Askell, A., Henighan, T., Drain, D., Perez, E., Schiefer, N., Hatfield-Dodds, Z., DasSarma, N., Tran-Johnson, E., et al.: Language models (mostly) know what they know. arXiv preprint arXiv:2207.05221 (2022)

[73] Kendall, A., Gal, Y.: What uncertainties do we need in bayesian deep learning for computer vision? In: Advances in neural information processing systems, pp. 5574–5584 (2017)

[74] Khosla, P., Teterwak, P., Wang, C., Sarna, A., Tian, Y., Isola, P., Maschinot, A., Liu, C., Krishnan, D.: Supervised contrastive learning. arXiv preprint arXiv:2004.11362 (2020)

[75] Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)

[76] Klein, A., Falkner, S., Springenberg, J.T., Hutter, F.: Learning curve prediction with Bayesian neural networks. International Conference on Learning Representations (2017)

[77] Kolagar, Z., Zarcone, A.: Aligning Uncertainty: Leveraging LLMs to Analyze Uncertainty Transfer in Text Summarization. In: Proceedings of the 1st Workshop on Uncertainty-Aware NLP (UncertaiNLP 2024), pp. 41–61 (2024)

[78] Kuhn, L., Gal, Y., Farquhar, S.: Semantic uncertainty: Linguistic invariances for uncertainty estimation in natural language generation. arXiv preprint arXiv:2302.09664 (2023)

[79] Kumar, A., Liang, P.S., Ma, T.: Verified uncertainty calibration. In: Advances in Neural Information Processing Systems, pp. 3787–3798 (2019)

[80] Laban, P., Schnabel, T., Bennett, P.N., Hearst, M.A.: SummaC: Re-visiting NLI-based models for inconsistency detection in summarization. Transactions of the Association for Computational Linguistics **10**, 163–177 (2022)

[81] Laine, S., Aila, T.: Temporal ensembling for semi-supervised learning. arXiv preprint arXiv:1610.02242 (2016)

[82] Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K., Dyer, C.: Neural architectures for named entity recognition. arXiv preprint arXiv:1603.01360 (2016)

[83] Lang, K.: Newsweeder: Learning to filter netnews. In: Machine Learning Proceedings 1995, pp. 331–339. Elsevier (1995)

[84] Lee, K., Lee, K., Lee, H., Shin, J.: A simple unified framework for detecting out-of-distribution samples and adversarial attacks. In: Advances in Neural Information Processing Systems, pp. 7167–7177 (2018)

[85] Lei, S., Zhang, X., He, J., Chen, F., Du, B., Lu, C.T.: Cross-domain few-shot semantic segmentation. In: European Conference on Computer Vision, pp. 73–90. Springer (2022)

[86] Lei, S., Zhang, X., He, J., Chen, F., Lu, C.T.: Few-Shot Semantic Segmentation via Prototype Augmentation with Image-Level Annotations. In: 2021 IEEE International Conference on Multimedia and Expo (ICME), pp. 1–6. IEEE (2021)

[87] Lei, S., Zhang, X., He, J., Chen, F., Lu, C.T.: Uncertainty-Aware Cross-Lingual Transfer with Pseudo Partial Labels. In: Findings of the Association for Computational Linguistics: NAACL 2022, pp. 1987–1997 (2022)

[88] Lei, S., Zhang, X., He, J., Chen, F., Lu, C.T.: TART: Improved Few-shot Text Classification Using Task-Adaptive Reference Transformation. In: Proceedings of the 61th Annual Meeting of the Association for Computational Linguistics (2023)

[89] Lei, Y., Song, K., Cho, S., Wang, X., Huang, R., Yu, D.: Polarity Calibration for Opinion Summarization. arXiv preprint arXiv:2404.01706 (2024)

[90] Lewis, D.D., Yang, Y., Russell-Rose, T., Li, F.: Rcv1: A new benchmark collection for text categorization research. Journal of machine learning research **5**(Apr), 361–397 (2004)

[91] Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., Zettlemoyer, L.: Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. arXiv preprint arXiv:1910.13461 (2019)

[92] Li, S., Zang, Z., Li, S.Z.: Exploring Localization for Self-supervised Fine-grained Contrastive Learning (2022)

[93] Liang, B., Li, H., Su, M., Bian, P., Li, X., Shi, W.: Deep text classification can be fooled. arXiv preprint arXiv:1704.08006 (2017)

[94] Lin, C.Y.: Rouge: A package for automatic evaluation of summaries. In: Text summarization branches out, pp. 74–81 (2004)

[95] Lin, Z., Trivedi, S., Sun, J.: Generating with confidence: Uncertainty quantification for black-box large language models. arXiv preprint arXiv:2305.19187 (2023)

[96] Liu, J., Lin, Z., Padhy, S., Tran, D., Bedrax Weiss, T., Lakshminarayanan, B.: Simple and Principled Uncertainty Estimation with Deterministic Deep Learning via Distance Awareness. In: H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, H. Lin (eds.) Advances in Neural Information Processing Systems, vol. 33, pp. 7498–7512. Curran Associates, Inc. (2020). URL https://proceedings.neurips.cc/paper_files/paper/2020/file/543e83748234f7cbab21aa0ade66565f-Paper.pdf

[97] Liu, J., Paisley, J., Kioumourtzoglou, M.A., Coull, B.: Accurate Uncertainty Estimation and Decomposition in Ensemble Learning. In: Advances in Neural Information Processing Systems, pp. 8950–8961 (2019)

[98] Liu, M., Tu, Z., Zhang, T., Su, T., Xu, X., Wang, Z.: Ltp: A new active learning strategy for crf-based named entity recognition. Neural Processing Letters **54**(3), 2433–2454 (2022)

[99] Liu, P., Fu, J., Xiao, Y., Yuan, W., Chang, S., Dai, J., Liu, Y., Ye, Z., Neubig, G.: ExplainaBoard: An Explainable Leaderboard for NLP. In: Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: System Demonstrations, pp. 280–289. Association for Computational Linguistics, Online (2021). DOI 10.18653/v1/2021.acl-demo.34. URL https://aclanthology.org/2021.acl-demo.34

[100] Liu, R., Azabou, M., Dabagia, M., Lin, C.H., Gheshlaghi Azar, M., Hengen, K., Valko, M., Dyer, E.: Drop, swap, and generate: A self-supervised approach for generating neural activity. Advances in neural information processing systems **34**, 10587–10599 (2021)

[101] Liu, Y.: Fine-tune BERT for extractive summarization. arXiv preprint arXiv:1903.10318 (2019)

[102] Liu, Y., Lapata, M.: Text summarization with pretrained encoders. arXiv preprint arXiv:1908.08345 (2019)

[103] Louizos, C., Welling, M.: Multiplicative normalizing flows for variational bayesian neural networks. In: Proceedings of the 34th International Conference on Machine Learning-Volume 70, pp. 2218–2227. JMLR. org (2017)

[104] Maddox, W.J., Izmailov, P., Garipov, T., Vetrov, D.P., Wilson, A.G.: A simple baseline for bayesian uncertainty in deep learning. Advances in Neural Information Processing Systems **32** (2019)

[105] Malinin, A., Gales, M.: Predictive uncertainty estimation via prior networks. Advances in neural information processing systems **31** (2018)

[106] Malinin, A., Gales, M.: Reverse kl-divergence training of prior networks: Improved uncertainty and adversarial robustness. In: Advances in Neural Information Processing Systems, pp. 14520–14531 (2019)

[107] Malinin, A., Gales, M.: Uncertainty estimation in autoregressive structured prediction. arXiv preprint arXiv:2002.07650 (2020)

[108] Malinin, A., Ragni, A., Knill, K., Gales, M.: Incorporating Uncertainty into Deep Learning for Spoken Language Assessment. In: R. Barzilay, M.Y. Kan (eds.) Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), pp. 45–50. Association for Computational Linguistics, Vancouver, Canada (2017). DOI 10.18653/v1/P17-2008. URL `https://aclanthology.org/P17-2008`

[109] Malinin, A., Ragni, A., Knill, K., Gales, M.: Incorporating uncertainty into deep learning for spoken language assessment. In: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), pp. 45–50 (2017)

[110] McAuley, J., Leskovec, J.: Hidden factors and hidden topics: understanding rating dimensions with review text. In: Proceedings of the 7th ACM conference on Recommender systems, pp. 165–172 (2013)

[111] Medsker, L.R., Jain, L.: Recurrent neural networks. Design and Applications **5**(64-67), 2 (2001)

[112] Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in neural information processing systems, pp. 3111–3119 (2013)

[113] Minaee, S., Kalchbrenner, N., Cambria, E., Nikzad, N., Chenaghlu, M., Gao, J.: Deep Learning–based Text Classification: A Comprehensive Review. ACM Computing Surveys (CSUR) **54**(3), 1–40 (2021)

[114] Misra, R.: News category dataset (2018)

[115] Mukherjee, S., Awadallah, A.: Uncertainty-aware self-training for few-shot text classification. Advances in Neural Information Processing Systems **33** (2020)

[116] Mukhoti, J., Kirsch, A., van Amersfoort, J., Torr, P.H., Gal, Y.: Deep Deterministic Uncertainty: A New Simple Baseline. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 24384–24394 (2023)

[117] Nadeem, U., Bennamoun, M., Sohel, F., Togneri, R.: Learning-Based confidence estimation for Multi-modal classifier fusion. In: International Conference on Neural Information Processing, pp. 299–312. Springer (2019)

[118] Naeini, M.P., Cooper, G., Hauskrecht, M.: Obtaining well calibrated probabilities using bayesian binning. In: Twenty-Ninth AAAI Conference on Artificial Intelligence (2015)

[119] Nallapati, R., Zhou, B., Gulcehre, C., Xiang, B., et al.: Abstractive text summarization using sequence-to-sequence rnns and beyond. arXiv preprint arXiv:1602.06023 (2016)

[120] Narayan, S., Cohen, S.B., Lapata, M.: Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. arXiv preprint arXiv:1808.08745 (2018)

[121] Nguyen, M.T., Zuccon, G., Demartini, G., et al.: Loss-based active learning for named entity recognition. In: 2021 International Joint Conference on Neural Networks (IJCNN), pp. 1–8. IEEE (2021)

[122] Niculescu-Mizil, A., Caruana, R.: Predicting good probabilities with supervised learning. In: Proceedings of the 22nd international conference on Machine learning, pp. 625–632 (2005)

[123] Onan, A., Koruko**u**glu, S., Bulut, H.: Ensemble of keyword extraction methods and classifiers in text classification. Expert Systems with Applications **57**, 232–247 (2016)

[124] OpenAI: GPT-3.5. `https://platform.openai.com/docs/models/gpt-3-5-turbo` (2023)

[125] Osawa, K., Swaroop, S., Khan, M.E.E., Jain, A., Eschenhagen, R., Turner, R.E., Yokota, R.: Practical deep learning with Bayesian principles. Advances in neural information processing systems **32** (2019)

[126] Ovadia, Y., Fertig, E., Ren, J., Nado, Z., Sculley, D., Nowozin, S., Dillon, J., Lakshminarayanan, B., Snoek, J.: Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift. Advances in neural information processing systems **32** (2019)

[127] Papadopoulos, A.A., Rajati, M.R., Shaikh, N., Wang, J.: Outlier exposure with confidence control for out-of-distribution detection. arXiv preprint arXiv:1906.03509 (2019)

[128] Papineni, K., Roukos, S., Ward, T., Zhu, W.J.: Bleu: a method for automatic evaluation of machine translation. In: Proceedings of the 40th annual meeting of the Association for Computational Linguistics, pp. 311–318 (2002)

[129] Park, S., Park, J., Shin, S.J., Moon, I.C.: Adversarial dropout for supervised and semi-supervised learning. In: Thirty-Second AAAI Conference on Artificial Intelligence (2018)

[130] Patel, K., Beluch, W., Zhang, D., Pfeiffer, M., Yang, B.: On-manifold adversarial data augmentation improves uncertainty calibration. In: 2020 25th International Conference on Pattern Recognition (ICPR), pp. 8029–8036. IEEE (2021)

[131] Pennington, J., Socher, R., Manning, C.D.: Glove: Global vectors for word representation. In: Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), pp. 1532–1543 (2014)

[132] Plappert, M., Houthooft, R., Dhariwal, P., Sidor, S., Chen, R.Y., Chen, X., Asfour, T., Abbeel, P., Andrychowicz, M.: Parameter space noise for exploration. arXiv preprint arXiv:1706.01905 (2017)

[133] Reimers, N., Gurevych, I.: Sentence-bert: Sentence embeddings using siamese bert-networks. arXiv preprint arXiv:1908.10084 (2019)

[134] Ren, J., Luo, J., Zhao, Y., Krishna, K., Saleh, M., Lakshminarayanan, B., Liu, P.J.: Out-of-distribution detection and selective generation for conditional language models. In: The Eleventh International Conference on Learning Representations (2022)

[135] Rezende, D., Mohamed, S.: Variational inference with normalizing flows. In: International conference on machine learning, pp. 1530–1538. PMLR (2015)

[136] Riquelme, C., Tucker, G., Snoek, J.: Deep bayesian bandits showdown: An empirical comparison of bayesian deep networks for thompson sampling. arXiv preprint arXiv:1802.09127 (2018)

[137] Ritter, H., Kukla, M., Zhang, C., Li, Y.: Sparse Uncertainty Representation in Deep Learning with Inducing Weights. arXiv preprint arXiv:2105.14594 (2021)

[138] Rohekar, R.Y., Gurwicz, Y., Nisimov, S., Novik, G.: Modeling uncertainty by learning a hierarchy of deep neural connections. In: Advances in Neural Information Processing Systems, pp. 4246–4256 (2019)

[139] Sensoy, M., Kaplan, L., Kandemir, M.: Evidential deep learning to quantify classification uncertainty. Advances in neural information processing systems **31** (2018)

[140] Shen, A., Beck, D., Salehi, B., Qi, J., Baldwin, T.: Modelling Uncertainty in Collaborative Document Quality Assessment. In: Proceedings of the 5th Workshop on Noisy User-generated Text (W-NUT 2019), pp. 191–201 (2019)

[141] Shen, T., Zhou, T., Long, G., Jiang, J., Pan, S., Zhang, C.: Disan: Directional self-attention network for rnn/cnn-free language understanding. In: Thirty-Second AAAI Conference on Artificial Intelligence (2018)

[142] Simpson, E., Gao, Y., Gurevych, I.: Interactive text ranking with bayesian optimization: A case study on community qa and summarization. Transactions of the Association for Computational Linguistics **8**, 759–775 (2020)

[143] Snoek, J., Ovadia, Y., Fertig, E., Lakshminarayanan, B., Nowozin, S., Sculley, D., Dillon, J., Ren, J., Nado, Z.: Can you trust your model's uncertainty? Evaluating predictive uncertainty under dataset shift. In: Advances in Neural Information Processing Systems, pp. 13969–13980 (2019)

[144] Song, X., Huang, L., Xue, H., Hu, S.: Supervised prototypical contrastive learning for emotion recognition in conversation. arXiv preprint arXiv:2210.08713 (2022)

[145] Stadler, M., Charpentier, B., Geisler, S., Zügner, D., Günnemann, S.: Graph Posterior Network: Bayesian Predictive Uncertainty for Node Classification. Advances in Neural Information Processing Systems **34** (2021)

[146] Sun, K., Yu, J., Zhang, L., Dong, Z.: A convolutional neural network model based on improved softplus activation function. In: International Conference on Applications and Techniques in Cyber Intelligence ATCI 2019: Applications and Techniques in Cyber Intelligence 7, pp. 1326–1335. Springer (2020)

[147] Sun, S., Sun, Q., Zhou, K., Lv, T.: Hierarchical attention prototypical networks for few-shot text classification. In: Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP), pp. 476–485 (2019)

[148] Sun, Y., He, J., Cui, L., Lei, S., Lu, C.T.: Exploring the Deceptive Power of LLM-Generated Fake News: A Study of Real-World Detection Challenges. arXiv preprint arXiv:2403.18249 (2024)

[149] Sun, Y., He, J., Lei, S., Cui, L., Lu, C.T.: Med-MMHL: A Multi-Modal Dataset for Detecting Human-and LLM-Generated Misinformation in the Medical Domain. arXiv preprint arXiv:2306.08871 (2023)

[150] Tas, O., Kiyani, F.: A survey automatic text summarization. PressAcademia Procedia **5**(1), 205–213 (2007)

[151] Thulasidasan, S., Chennupati, G., Bilmes, J.A., Bhattacharya, T., Michalak, S.: On mixup training: Improved calibration and predictive uncertainty for deep neural networks. In: Advances in Neural Information Processing Systems, pp. 13888–13899 (2019)

[152] Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S., et al.: Llama 2: Open foundation and fine-tuned chat models. arXiv preprint arXiv:2307.09288 (2023)

[153] Tsvigun, A., Lysenko, I., Sedashov, D., Lazichny, I., Damirov, E., Karlov, V., Belousov, A., Sanochkin, L., Panov, M., Panchenko, A., et al.: Active learning for abstractive text summarization. arXiv preprint arXiv:2301.03252 (2023)

[154] Vasudevan, V.T., Sethy, A., Ghias, A.R.: Towards Better Confidence Estimation for Neural Models. In: ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 7335–7339. IEEE (2019)

[155] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, **L.**, Polosukhin, I.: Attention is all you need. In: Advances in neural information processing systems, pp. 5998–6008 (2017)

[156] Vazhentsev, A., Kuzmin, G., Shelmanov, A., Tsvigun, A., Tsymbalov, E., Fedyanin, K., Panov, M., Panchenko, A., Gusev, G., Burtsev, M., et al.: Uncertainty estimation of transformer predictions for misclassification detection. In: Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 8237–8252 (2022)

[157] Vijayakumar, A.K., Cogswell, M., Selvaraju, R.R., Sun, Q., Lee, S., Crandall, D., Batra, D.: Diverse beam search: Decoding diverse solutions from neural sequence models. arXiv preprint arXiv:1610.02424 (2016)

[158] Vinyals, O., Blundell, C., Lillicrap, T., Wierstra, D., et al.: Matching networks for one shot learning. Advances in neural information processing systems **29** (2016)

[159] Wang, D., Pandey, D.S., Neupane, K.P., Yu, Z., Zheng, E., Zheng, Z., Yu, Q.: Deep temporal sets with evidential reinforced attentions for unique behavioral pattern discovery. In: International Conference on Machine Learning, pp. 36205–36223. PMLR (2023)

[160] Wang, L., Lei, S., He, J., Wang, S., Zhang, M., Lu, C.T.: Self-Correlation and Cross-Correlation Learning for Few-Shot Remote Sensing Image Semantic Segmentation. In: Proceedings of the 31st ACM International Conference on Advances in Geographic Information Systems, pp. 1–10 (2023)

[161] Wang, S., Liu, Y., Wang, C., Luan, H., Sun, M.: Improving Back-Translation with Uncertainty-based Confidence Estimation. In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pp. 791–802 (2019)

[162] Wang, S., Sun, X., Li, X., Ouyang, R., Wu, F., Zhang, T., Li, J., Wang, G.: Gpt-ner: Named entity recognition via large language models. arXiv preprint arXiv:2304.10428 (2023)

[163] Wang, S., Tu, Z., Shi, S., Liu, Y.: On the Inference Calibration of Neural Machine Translation. arXiv preprint arXiv:2005.00963 (2020)

[164] Wang, X., Qi, G.J.: Contrastive learning with stronger augmentations. IEEE Transactions on Pattern Analysis and Machine Intelligence (2022)

[165] Wang, Y., Mukherjee, S., Chu, H., Tu, Y., Wu, M., Gao, J., Awadallah, A.H.: Meta self-training for few-shot neural sequence labeling. In: Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, pp. 1737–1747 (2021)

[166] Wang, Y., Wang, H., Shen, Y., Fei, J., Li, W., Jin, G., Wu, L., Zhao, R., Le, X.: Semi-supervised semantic segmentation using unreliable pseudo-labels. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 4248–4257 (2022)

[167] Wang, Z., Wang, Y., Dong, B., Pracheta, S., Hamlen, K., Khan, L.: Adaptive margin based deep adversarial metric learning. In: 2020 IEEE 6th Intl Conference on Big Data Security on Cloud (BigDataSecurity), IEEE Intl Conference on High Performance and Smart Computing,(HPSC) and IEEE Intl Conference on Intelligent Data and Security (IDS), pp. 100–108. IEEE (2020)

[168] Wen, Y., Vicol, P., Ba, J., Tran, D., Grosse, R.: Flipout: Efficient pseudo-independent weight perturbations on mini-batches. arXiv preprint arXiv:1803.04386 (2018)

[169] Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., Brew, J.: HuggingFace's Transformers: State-of-the-art Natural Language Processing. ArXiv **abs/1910.03771** (2019)

[170] Wong, K.F., Wu, M., Li, W.: Extractive summarization using supervised and semi-supervised learning. In: Proceedings of the 22nd international conference on computational linguistics (Coling 2008), pp. 985–992 (2008)

[171] Wu, X., Gao, C., Lin, Z., Han, J., Wang, Z., Hu, S.: InfoCSE: Information-aggregated Contrastive Learning of Sentence Embeddings. arXiv preprint arXiv:2210.06432 (2022)

[172] Xia, C., Xiong, C., Yu, P.: Pseudo siamese network for few-shot intent generation. In: Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 2005–2009 (2021)

[173] Xiao, Y., Wang, W.Y.: Quantifying uncertainties in natural language processing tasks. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, pp. 7322–7329 (2019)

[174] Xie, S., Kirillov, A., Girshick, R., He, K.: Exploring randomly wired neural networks for image recognition. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 1284–1293 (2019)

[175] Yan, Y., Li, R., Wang, S., Zhang, F., Wu, W., Xu, W.: ConSERT: A Contrastive Framework for Self-Supervised Sentence Representation Transfer. arXiv preprint arXiv:2105.11741 (2021)

[176] Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R.R., Le, Q.V.: Xlnet: Generalized autoregressive pretraining for language understanding. In: Advances in neural information processing systems, pp. 5754–5764 (2019)

[177] Yeh, C.H., Hong, C.Y., Hsu, Y.C., Liu, T.L., Chen, Y., LeCun, Y.: Decoupled contrastive learning. In: European Conference on Computer Vision, pp. 668–684. Springer (2022)

[178] Yoo, K., Kim, J., Jang, J., Kwak, N.: Detection of word adversarial examples in text classification: Benchmark and baseline via robust density estimation. arXiv preprint arXiv:2203.01677 (2022)

[179] Yu, L., Lou, Y., Chen, F.: Uncertainty-aware Graph-based Hyperspectral Image Classification. In: The Twelfth International Conference on Learning Representations (2023)

[180] Yuan, W., Neubig, G., Liu, P.: Bartscore: Evaluating generated text as text generation. Advances in Neural Information Processing Systems **34**, 27263–27277 (2021)

[181] Zablotskaia, P., Phan, D., Maynez, J., Narayan, S., Ren, J., Liu, J.: On uncertainty calibration and selective generation in probabilistic neural summarization: A benchmark study. arXiv preprint arXiv:2304.08653 (2023)

[182] Zaidi, S., Zela, A., Elsken, T., Holmes, C., Hutter, F., Teh, Y.W.: Neural Ensemble Search for Uncertainty Estimation and Dataset Shift. Advances in Neural Information Processing Systems **34** (2021)

[183] Zhang, H., Cisse, M., Dauphin, Y.N., Lopez-Paz, D.: mixup: Beyond empirical risk minimization. arXiv preprint arXiv:1710.09412 (2017)

[184] Zhang, L., Ma, B., He, J., Li, G., Huang, Q., Tian, Q.: Adaptively Unified Semi-supervised Learning for Cross-Modal Retrieval. In: IJCAI, pp. 3406–3412 (2017)

[185] Zhang, M., He, J., Ji, T., Lu, C.T.: Don't Go To Extremes: Revealing the Excessive Sensitivity and Calibration Limitations of LLMs in Implicit Hate Speech Detection. arXiv preprint arXiv:2402.11406 (2024)

[186] Zhang, M., He, J., Lei, S., Yue, M., Wang, L., Lu, C.T.: Can LLM find the green circle? Investigation and Human-guided tool manipulation for compositional generalization. In: ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 11996–12000. IEEE (2024)

[187] Zhang, R., Tetreault, J.: This email could save your life: Introducing the task of email subject line generation. arXiv preprint arXiv:1906.03497 (2019)

[188] Zhang, T., Kishore, V., Wu, F., Weinberger, K.Q., Artzi, Y.: Bertscore: Evaluating text generation with bert. arXiv preprint arXiv:1904.09675 (2019)

[189] Zhang, X., Chen, F., Lu, C.T., Ramakrishnan, N.: Mitigating Uncertainty in Document Classification. In: Proceedings of NAACL-HLT, pp. 3126–3136 (2019)

[190] Zhang, X., He, S., Liu, K., Zhao, J.: AdaNSP: Uncertainty-driven Adaptive Decoding in Neural Semantic Parsing. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pp. 4265–4270 (2019)

[191] Zhang, X., Liu, Y., Wang, X., He, P., Yu, Y., Chen, S.Q., Xiong, W., Wei, F.: Momentum calibration for text generation. arXiv preprint arXiv:2212.04257 (2022)

[192] Zhang, X., Zhao, J., LeCun, Y.: Character-level convolutional networks for text classification. In: Advances in neural information processing systems, pp. 649–657 (2015)

[193] Zhang, Z., He, J., Kumar, A., Rahman, S.: AI-based Space Occupancy Estimation Using Environmental Sensor Data. In: 2024 IEEE Power & Energy Society Innovative Smart Grid Technologies Conference (ISGT), pp. 1–5. IEEE (2024)

[194] Zhang, Z., Hu, M., Zhao, S., Huang, M., Wang, H., Liu, L., Zhang, Z., Liu, Z., Wu, B.: E-NER: Evidential Deep Learning for Trustworthy Named Entity Recognition. arXiv preprint arXiv:2305.17854 (2023)

[195] Zhao, X., Chen, F., Hu, S., Cho, J.H.: Uncertainty aware semi-supervised learning on graph data. Advances in Neural Information Processing Systems **33**, 12827–12836 (2020)

[196] Zhao, Y., Khalman, M., Joshi, R., Narayan, S., Saleh, M., Liu, P.J.: Calibrating sequence likelihood improves conditional language generation. In: The Eleventh International Conference on Learning Representations (2022)

[197] Zhong, M., Liu, Y., Yin, D., Mao, Y., Jiao, Y., Liu, P., Zhu, C., Ji, H., Han, J.: Towards a unified multi-dimensional evaluator for text generation. arXiv preprint arXiv:2210.07197 (2022)

[198] Zoph, B., Le, Q.V.: Neural architecture search with reinforcement learning. arXiv preprint arXiv:1611.01578 (2016)