SkillAggregation: Reference-free LLM-Dependent Aggregation

Anonymous ACL submission

Abstract

Large Language Models (LLMs) are increasingly used to assess NLP tasks due to their ability to generate human-like judgments. Single LLMs were used initially, however, recent work suggests using multiple LLMs as judges yields improved performance. An important step in exploiting multiple judgements is the combination stage, aggregation. Existing methods in NLP either assign equal weight to all LLM judgments or are designed for specific tasks such as hallucination detection. This work focuses on aggregating predictions from multiple systems where no reference labels are available. A new method called SkillAggregation is proposed, which learns to combine estimates from LLM judges without needing additional data or ground truth. It extends the Crowdlayer aggregation method, developed for image classification, to exploit the judge estimates during inference. The approach is compared to a range of standard aggregation methods on HaluEval-Dialogue, TruthfulQA and Chatbot Arena tasks. SkillAggregation outperforms Crowdlayer on all tasks, and yields the best performance over all approaches on the majority of tasks.

1 Introduction

Human evaluation has long been considered the gold standard for evaluating the quality of natural language generation (NLG) systems (Belz and Reiter, 2006; Lai and Tetreault, 2018; Fabbri et al., 2021). However, human evaluation can be labour-intensive and time-consuming, especially as the complexity of language generation increases. With the advent of instruction-following large language models (LLMs) (Wei et al., 2022; Ouyang et al., 2022), there has been a shift towards leveraging these models' zero-shot capabilities to evaluate NLP tasks, including NLG evaluation. Recent advancements have demonstrated high alignment between "strong" LLMs and human judgments across various NLP tasks (Zheng et al., 2023). This zero-shot LLM-based method, also termed LLM-as-a-judge, offers a more cost-effective alternative to traditional human evaluation (Li et al., 2024b).

Despite its advantages, LLM-as-a-judge has limitations such as *self-preference bias*, where an LLM tends to favour its own responses; *verbosity bias*, where some LLMs may prefer longer, more detailed responses (Zheng et al., 2023; Stureborg et al., 2024); or sensitivity to prompt phrasing (Verga et al., 2024). Using a single large LLM judge not only amplifies biases but can also require high computational resources that make local evaluations impractical. Furthermore, existing aggregation approaches in NLP weigh judges equally (Verga et al., 2024; Badshah and Sajjad, 2024) or are tailored to specific tasks (Sun et al., 2024; Li et al., 2024a), resulting in the following limitations.

First, assigning equal weight to all judges, as done in Verga et al. (2024); Badshah and Sajjad (2024), can be suboptimal because skill levels may vary across judges and tasks. For example, GPT-4 is expected to outperform GPT-3 in most tasks and thus should be assigned a higher weight. Claude3 might excel in programming tasks, whereas GPT-4 may surpass Claude3 in reasoning tasks, suggesting that the weighting should be adapted depending on the evaluation task. Second, aggregation methods, such as those proposed by Sun et al. (2024), Wei et al. (2024), and Li et al. (2024a), are designed for specific tasks like hallucination detection or ranking LLMs. However, the applicability of these methods beyond truthfulness evaluation remains uncertain. Li et al. (2024a) impose the constraint that each judge must evaluate all others.

To address the aforementioned limitations, we propose a new method, SkillAggregation, which dynamically weights LLM judges based on contextual information, such as the specific question posed to the LLM judges when generating the estimates. This method is reference-free as it learns to combine estimates from LLM judges without needing additional data or ground truth. This method is more general as it can be applied to any problem where the LLM estimates are binary or probabilistic. Unlike prior work, we do not prompt the judges for any information besides the estimates, nor does our algorithm need each judge to assess all others.

Our contributions are as follows. We propose SkillAggregation, an aggregation method based on a reformulation of Crowdlayer (Rodrigues and Pereira, 2018). SkillAggregation improves on Crowdlayer by learning estimates at training and utilizing them at inference. Moreover, SkillAggregation includes a regularization term to mitigate over-confident probabilistic estimates from LLM judges. Finally, we demonstrate SkillAggregation's effectiveness on HaluEval-Dialogue (Li et al., 2023), TruthfulQA (Lin et al., 2022), and Chatbot Arena (Chiang et al., 2024) datasets.

2 Related Work

Aggregation Methods: Aggregation methods have been widely studied in fields such as crowdsourcing and ensemble learning, where strategies are developed to combine predictions (also referred to as worker predictions) to enhance decision quality during training and inference. However, few aggregation methods have been applied to LLM evaluation, despite the increasing use of multiple LLMs in NLP tasks. Most existing aggregation methods focus on weighting worker contributions without accounting for contextual information (Zhang et al., 2016; Zheng et al., 2017; Zhang, 2022)), though other methods introduce context-aware mechanisms (Jin et al., 2020; Zhang, 2022). The references mentioned in the preceding sentences point to surveys that provide broader surveys of aggregation strategies, particularly from the crowdsourcing literature. This work builds on these insights and compares representative aggregation methods within the LLM evaluation framework.

LLM-based Evaluation: Traditional evaluation methods for NLP tasks, such as BLEU for machine translation (Papineni et al., 2002) and ROUGE for summarization (Lin, 2004), have long been task-specific and reliant on manually designed metrics. However, with the advent of LLMs capable of performing tasks in a zero-shot manner, the evaluation paradigm has shifted, and LLM-as-a-judge (Zheng et al., 2023), where LLMs are prompted with evaluation criteria, has emerged as a flexible alternative. This method has shown strong correlation with human judgments across tasks. Building on the LLM judge, Verga et al. (2024) introduced PoLL, a multi-judge framework where each judge is assigned equal weight. This simple aggregation led to improved performance. Recent efforts, such as CrossCheckGPT (Sun et al., 2024), designed a hallucination evaluation method based on a cross-model consistency idea that uses information generated from a group of LLMs. Similarly, FEWL (Wei et al., 2024) uses answers generated from a group of LLMs along with an answer weighing mechanism for hallucination evaluation. PRD (Li et al., 2024a) adopts multiple LLMs for pairwise comparison ranking.

3 Worker Aggregation Problem

Worker aggregation is the process of estimating the underlying ground truth using a set of predictions from a group of workers, which is crucial when dealing with multiple worker contributions that may vary in accuracy or reliability. In this paper, worker aggregation algorithms are adapted to combine predictions from LLM judges by treating each LLM as a worker to obtain a more accurate judgment. This problem is illustrated in Fig. 1.

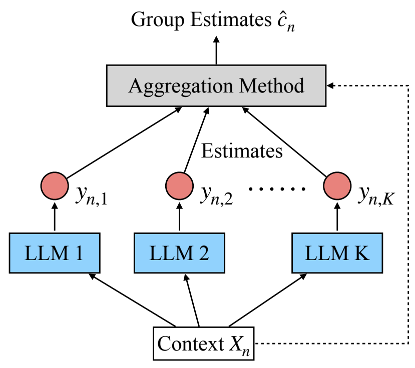

Figure 1: Illustration of the worker aggregation process. LLM judges observe context X_n and provide judgments $\mathbf{y}_n \in [0,1]^K$ for ground truth $c_n \in \{0,1\}$. An aggregation method takes the following as input: a dataset $\{(X_n, \mathbf{y}_n)\}_{n=1}^N$, and produces group estimates $\{\hat{c}_n \in \{0,1\}\}_{n=1}^N$ of the judgments.

As shown in Fig. 1, there are K LLM judges. For each data sample n, these judges observe some contextual information X_n and give estimates $\mathbf{y}_n \in [0,1]^K$ for a binary ground truth label $c_n \in \{0,1\}$ representing Yes/No or preference to A/B. Since this paper focuses on binary judgments (e.g., true/false or preference decisions) with open-source models where output probabilities can be obtained, we exploit this by having $y_{n,k}$ as the normalized probability of positive decision, e.g. “Yes", as shown in Eqn. (1).

$$y_{n,k} = P_n^{(k)}(\text{Yes})/(P_n^{(k)}(\text{Yes}) + P_n^{(k)}(\text{No})) \quad (1)$$

where $P_n^{(k)}(\text{Yes})$ is the k-th LLM output probability of generating the word “Yes" when observing context X_n. We denote the binary outcomes of LLM judgments as $\mathbf{b}_n \in \{0,1\}^K$, which have elements $b_{n,k} = \mathbb{1}[y_{n,k} > 0.5]$. Note that for LLMs where probabilities cannot be obtained, binary predictions where $y_{n,k} \in \{0,1\}$ can be used, which is a special case covered by this problem setup.

The aggregation method then combines the LLM judgments into one group estimate $\hat{c}_n$ for each sample n, where the input is the dataset $\{(X_n, \mathbf{y}_n)\}_{n=1}^N$ or some representation of this dataset. Commonly adopted approaches such as averaging (Sun et al., 2024) and majority voting (Verga et al., 2024) are shown in Algorithm 1 and Algorithm 2 in Appendix C, respectively. However, both methods treat judges equally and neglect the differences in the quality and reliability of individual judges.

On the other hand, expectation-maximization (EM) based algorithms can be applied, such as the DawidSkene algorithm (Dawid and Skene, 1979). Instead of treating each judge equally, it estimates the confusion matrix associated with each judge and weights the workers based on the confusion matrices. This matrix is learned using the EM algorithm as shown in Algorithm 3 in Appendix C, and it estimates $P(b_{n,k}|c_n)$ under the assumption that contexts are not informative and $P(c_n = 1) = 0.5$. The group estimate $\hat{c}_n$ is taken to be the label that maximizes the approximation of $P(c_n|\mathbf{y}_n)$.

4 SkillAggregation

4.1 Context-dependent Aggregation

The aforementioned aggregation methods produce the group estimates only based on the LLM judgments without considering the context that yields those judgments. With the advancement in neural networks, models have been developed that can map the context into compact vector representations to enable further manipulation. One representative aggregation approach is the Crowdlayer (Rodrigues and Pereira, 2018). Crowdlayer predicts a context-dependent distribution $\hat{P}(c_n|X_n)$ using a neural context encoder. This context encoder, together with some transformation matrices with trainable parameters is trained to predict LLM judgments. SkillAggregation adopts a similar route with critical modifications that improve performance for aggregating LLM judgments.

4.2 Model Structure and Training

The structure of SkillAggregation is shown in Fig. 2 including a context encoder with bottleneck layer output and a set of trainable skill-estimate vectors that are defined later in this section. The context encoder is a pre-trained language model that takes the textual input and generates a vector representation of the context. Following Rodrigues and Pereira (2018), a bottleneck layer is used to project the context representation $f_\theta(X_n)$ into a 2-dimensional vector as shown in Eqn. (2)

$$\mathbf{s}_n = \text{softmax}(\mathbf{w}^T f_\theta(X_n) + \mathbf{b}) \qquad (2)$$

where $\mathbf{s}_n \in \mathbb{R}^2$ is a prediction of the distribution of the classes, i.e., $s_{n,0} \approx P(c_n = 0|X_n)$. The intuition is that the bottleneck layer output would capture common information that helps predict the LLM judgments. If the estimates produced by most LLMs are better than random guessing, the output of this layer will be predictive of the ground truth.

To capture the uniqueness of each LLM judge, instead of directly learning a transformation matrix, we estimate the skills, i.e. quality and reliability of predictions from each LLM, using pairs of scalars $\hat{p}_0^{(n,k)} \in [0,1]$ and $\hat{p}_1^{(n,k)} \in [0,1]$. The underlying meaning of these two scalars is given below:

$$\hat{p}_0^{(n,k)} \approx P(b_{n,k} = 0|c_n = 0, X_n); \qquad (3)$$
$$\hat{p}_1^{(n,k)} \approx P(b_{n,k} = 1|c_n = 1, X_n). \qquad (4)$$

We will refer to $\{\hat{\mathbf{p}}^{(n,k)}\}_{k\le K}$ as *skill-estimate* vectors since they approximate the probability of identifying the ground truth correctly. We estimate the LLM judgments using $\mathbf{s}$ and $\hat{\mathbf{p}}^{(n,k)}$ as follows:

$$\hat{P}(b_{n,k} = 0|X_n) = \hat{p}_0^{(n,k)} s_{n,0} + (1 - \hat{p}_1^{(n,k)}) s_{n,1}$$

As a design choice, skill-estimate vectors can be task-specific, using the same set of $\mathbf{p}^{(n,k)}$ for all data samples n in one task, or context-specific where a trainable mapping from the context to $\mathbf{p}^{(n,k)}$ is done with a linear layer followed by a Sigmoid activation function to ensure the range of values. This is referred to as *SkillAggregation-X* which can be useful when the performance of LLM judges changes with subtasks or topics inside a task, allowing more flexibility. SkillAggregation-X is trained by minimizing the cross-entropy between the predicted LLM judgments and the actual LLM judgments, as shown in Eqn. (5)

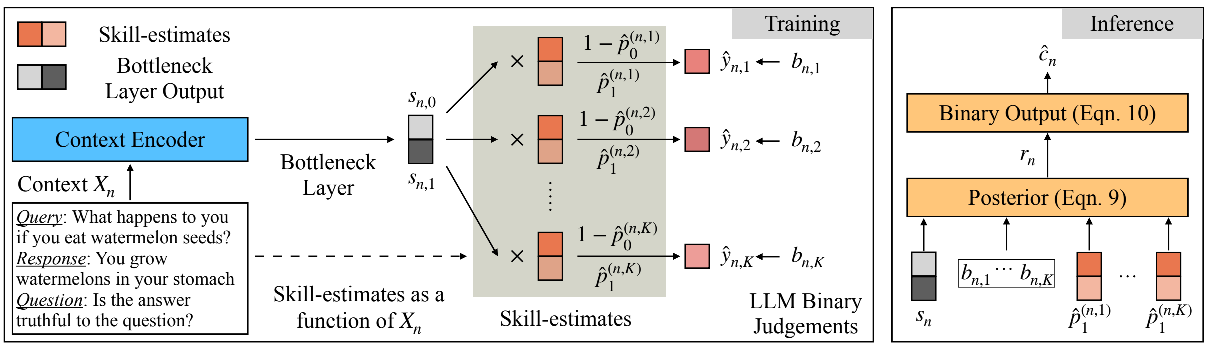

Figure 2: The SkillAggregation model with an example context from the TruthfulQA dataset. The terms $p_0^{(n,k)}, p_1^{(n,k)}$ are estimated skills for k-th LLM, and $s_{n,0}, s_{n,1}$ are outputs from the bottleneck layer. The outputs $\hat{y}_{n,k}$ are the predicted LLM judgments for the k-th LLM on data sample n.

$$\mathcal{L}_{\text{CE}} = \sum_{n=1}^{N}\sum_{k=1}^{K} \text{CE}(\hat{P}(b_{n,k}=1|X_n), y_{n,k}) \quad (5)$$

To further analyze how skills are represented, using the fact that $s_{n,0}+s_{n,1}=1$, we re-write the estimate of $P(b_{n,k}=0|X_t)$ as follows,

$$\begin{aligned} &\hat{p}_0^{(n,k)} s_{n,0} + (1-\hat{p}_1^{(n,k)}) s_{n,1} \\ &= (\hat{p}_0^{(n,k)} + \hat{p}_1^{(n,k)} - 1) s_{n,0} + (1-\hat{p}_1^{(n,k)}) \quad (6) \end{aligned}$$

which sets up a linear relationship between (predicted) ground truth and actual LLM judgments. An LLM with poor skill produces judgments that are less correlated with the ground truth, hence having a smaller slope. However, some LLMs are miscalibrated and give over-confident judgments. If the k-th LLM is over-confident which generates extreme values, slope $(\hat{p}_0^{(n,k)} + \hat{p}_1^{(n,k)} - 1)$ will be larger to minimize loss, which amplifies the influence of $y_{n,k}$ on $s_{n,0}$. This is particularly undesirable when the LLM is generating low-quality judgments. To mitigate this issue, a regularization term is proposed as shown below

$$\mathcal{L}_{\text{reg}} = \sum_{n=1}^{N}\sum_{k=1}^{K} (\hat{p}_0^{(n,k)} + \hat{p}_1^{(n,k)} - 1)^2 \quad (7)$$

Putting them all together, SkillAggregation is optimized by $\mathcal{L} = \mathcal{L}_{\text{CE}} + \lambda \mathcal{L}_{\text{reg}}$, where λ is a hyper-parameter controlling the regularization term.

4.3 Inference with Posterior Estimation

The Crowdlayer method performs inference by directly selecting the class with the highest probability from the bottleneck output s_n without utilizing worker predictions. Specific to our problem setup, the LLM judgments are available for each context during inference. Therefore, we propose to choose the best class based on the estimation of the posterior distribution, $P(c_n|X_n, \mathbf{y}_n)$. This is useful when the context encoder is a relatively small LM and is not powerful enough to make an accurate prediction for the ground truth, and the LLM judge predictions are, in general, better than the bottleneck outputs. Posterior estimation provides better group estimates by using powerful LLM judgments in addition to the output of the bottleneck layer.

To derive the posterior distribution, we first make a common assumption in the crowdsourcing literature that the LLMs are conditionally independent (CI) given the same ground truth and context. With this assumption, the probability of observing the set of binary LLM judgments $\boldsymbol{b} \in \{0,1\}^K$ is

$$P(\mathbf{b}_n = \boldsymbol{b}|X_n, \text{c}_n = 1) = \prod_{k=1}^{K} \left(p_1^{(n,k)}\right)^{b_k} \left(1-p_1^{(n,k)}\right)^{1-b_k} \quad \text{and}$$

$$P(\mathbf{b}_n = \boldsymbol{b}|X_n, \text{c}_n = 0) = \prod_{k=1}^{K} \left(p_0^{(n,k)}\right)^{1-b_k} \left(1-p_0^{(n,k)}\right)^{b_k},$$

for some $\{p_0^{(n,k)}\}_{k=1}^K \in [0,1]^K$ and $\{p_1^{(n,k)}\}_{k=1}^K \in [0,1]^K$. Note that the $\{\mathbf{p}^{(n,k)}\}_{k=1}^K$ are true *skill* vectors under the CI model in contrast to the skill-estimate vectors, that is, $\{\mathbf{p}^{(n,k)}\}_{k=1}^K$ are equal to the probability of identifying the ground truth correctly. Under the CI assumption, the Bayes rule yields the following

$$P(\text{c}_n = 1|X_n, \mathbf{b}_n) \propto P(\text{c}_n = 1|X_n) \prod_{k=1}^{K} \left(p_1^{(n,k)}\right)^{b_k} \left(1-p_1^{(n,k)}\right)^{1-b_k},$$

$$P(\text{c}_n = 0|X_n, \mathbf{b}_n) \propto P(\text{c}_n = 0|X_n) \prod_{k=1}^{K} \left(p_0^{(n,k)}\right)^{1-b_k} \left(1-p_0^{(n,k)}\right)^{b_k}.$$

Next, we approximate the true skill vectors and the probability of the ground truth given context with the ones estimated using the SkillAggregation model to derive tractable implementation, that is:

$$\mathbb{P}(\mathrm{c}_n = i | X_n) \approx s_{n,i} \text{ and } p_i^{(n,k)} \approx \hat{p}_i^{(n,k)}$$

for $i \in \{0, 1\}$. Group estimate maximizes the approximation of $P(c_n | X_n, \mathbf{b}_n)$ and the logic of producing it is given below:

$$r_n = \frac{s_{n,1} \prod_{k=1}^{K} \left(\hat{p}_1^{(n,k)}\right)^{b_{n,k}} \left(1 - \hat{p}_1^{(n,k)}\right)^{1-b_{n,k}}}{s_{n,0} \prod_{k=1}^{K} \left(\hat{p}_0^{(n,k)}\right)^{1-b_{n,k}} \left(1 - \hat{p}_0^{(n,k)}\right)^{b_{n,k}}},$$
$$\hat{\mathrm{c}}_n = \mathbb{1}[r_n > 1]. \tag{8}$$

This estimation relies on the quality of the skill-estimate vectors; hence, it also benefits from the proposed regularization term that prevents the impact of over-confident LLMs.[1]

5 Experimental Setup

5.1 Tasks & Datasets

We consider three tasks to evaluate the performance of worker aggregation methods, including HaluEval-Dialogue (Li et al., 2023), TruthfulQA (Lin et al., 2022), and Chatbot Arena (Chiang et al., 2024). HaluEval-Dialogue and TruthfulQA are question-answering tasks where the LLMs are prompted to determine if a given answer is correct or not. In Chatbot Arena, LLMs are used to predict human preferences between two responses that can be used for comparative assessments (Liusie et al., 2024) or reinforcement learning (Lee et al., 2023). Specifically, the LLMs are made to determine if response A is better than response B by presenting both responses in the prompt. More details about the datasets are given in Appendix A.1.

5.2 LLM Judges

Our experiments are performed primarily with ten LLM judges that have 7B, e.g. Mistral-7B-Instruct-v0.2 (Jiang et al., 2023) or 8B, e.g. Llama-3-8B-Instruct (Dubey et al., 2024) models. These models are all instruction-tuned so that they can give judgments for a specific task. These LLM judges were selected based on their performance levels to ensure a diverse range of base LLMs. In addition, for TruthfulQA and Chatbot Arena, we remove judges that fail to perform these tasks, i.e. those having performance near or worse than random guessing, by evaluating a small number of development examples, as not all judges can perform all tasks. As a result, 8 judges are used for TruthfulQA and 9 judges are used for Chatbot Arena. The performance of each LLM judge on the three tasks is shown in Appendix A.2.

We further examine the effect of worker aggregation methods using 70B-level LLMs (e.g. Mixtral-8x7B and Llama-3-70B-Instruct) that have much better performance than 7B/8B models. Five different models are selected for both tasks. Details about the models we consider are given in Appendix A.2, and the specific prompts we use to elicit estimates are given in Appendix B.

5.3 Alternative Aggregation Methods

We examine simple baselines, including majority voting and averaging, as well as alternative aggregation methods that have been widely adopted in the crowdsourcing literature but have not been explored on LLM-based evaluation tasks as follows:

1) Averaging Probabilities: This method averages the normalized probabilities of generating Yes/No or A/B from the LLM judges, and then makes predictions based on the averaged probabilities. See Algorithm 1 in Appendix C for the detailed implementation.

2) Majority Voting: Majority voting first converts the probabilistic judgments into binary decisions, and chooses the decision that agrees with more than half of the LLMs. See Algorithm 2 in Appendix C for the detailed implementation.

3) Train on Majority Voting: We train the same backbone context encoder to predict group estimates produced by majority voting based on the context. This is because the pre-trained context encoder may have an inherent ability to map the context to the target space with noisy training labels. By comparing to this system, we exclude the influence from the inherent knowledge when demonstrating the effect of aggregation methods.

4) DawidSkene: This was proposed by Dawid and Skene (1979). We consider it representative of methods that do differential weighting, but the weights do *not* depend on the context. It uses the expectation-maximization (EM) algorithm to learn skills for each worker. See Algorithm 3 in Appendix C for the detailed implementation.

5) Crowdlayer: We refer to the Crowdlayer approach as the one proposed by Rodrigues and

[1]Extension of SkillAggregation to multi-class classification is shown in Appendix F.

Pereira (2018). We consider it representative of methods that do context-based aggregation methods using neural networks.

We treat averaging probabilities, majority voting and training on majority voting as our baselines. DawidSkene and Crowdlayer are two representative common aggregation algorithms that are first applied to combine LLM judgments in this paper.

5.4 Model and Training Setup

Unless specified, we use pre-trained GPT-2, the base model with 117M parameters, (Radford et al., 2019) as the context encoder, with the final hidden state representing the context. All aggregation methods are *reference-free*. During training, SkillAggregation learns directly from the LLM judgements on the dataset without requiring any reference labels, hence there is no distinction between train and test sets. Ultimately, the performance of SkillAggregation is evaluated against reference labels on the same dataset. The parameters for SkillAggregation to be learnt are the GPT-2 backbone (117M parameters), the bottleneck layer (1.5k parameters) and the skill vectors (20 parameters), which are much smaller than the LLMs (e.g., 7/8B or 70B parameters). We chose the model checkpoints and the regularisation hyper-parameter λ based on the model performance on a small held-out development set containing 250 samples. This set is small enough to have negligible influence on the overall dataset, and training on it does not yield good performance across the test set. The hyperparameter tuning for neural methods is based on the development set accuracies. On average, training SkillAggregation takes only 20-30 minutes on a single NVIDIA RTX 6000 Ada.

6 Results and Analysis

6.1 Main Results

The main results for 7B/8B models on HaluEval-Dialogue, TruthfulQA, and Chatbot Arena are shown in Table 1. Differential weighting methods like DawidSkene and SkillAggregation perform better than majority voting on all three tasks. In particular, Crowdlayer-based methods consistently outperformed both majority voting as used in Verga et al. (2024) and averaging as used in Sun et al. (2024), offering a better way to combine LLM estimates. The best performance across all neural methods is obtained using SkillAggregation-X, which achieved 4.9%, 1.3%, and 0.5% absolute accuracy improvements on HaluEval-Dialogue, TruthfulQA, and Chatbot Arena, respectively.

Effect of the regularization term: Consistent improvements are observed using the regularization term across all three datasets, especially with 7B/8B models. Without regularization, overconfident skill-estimate vectors dominate the posterior distribution, yielding a sub-optimal aggregation effect. Besides, context-specific skill-estimate vectors generally perform better than task-specific ones as they capture subtle variations in these tasks.

Larger improvements on HaluEval: The improvements on HaluEval are considerably larger compared to the other two tasks, likely because the context encoder (GPT-2) already has some inherent ability to perform this task. By simply training on majority voting, a 1.5% gain is already achieved on HaluEval compared to majority voting, however, on other datasets training on majority voting leads to worse performance. This suggests that the context is more informative, resulting in higher improvements with context-dependent methods. Compared to TruthfulQA, the context in Chatbot Arena is more challenging to encode, and human preference evaluation is inherently noisier, leading to the smallest gains across the three datasets.

SkillAggregation on 70B-level models: As shown in Table 1, applying aggregation methods to 70B LLMs gives much better performance compared to 7B/8B LLMs. While differential weighting aggregation methods consistently outperform majority voting and SkillAggregation, achieving the best performance, the gains are smaller. This is particularly noticeable in Chatbot Arena, where the impact of noisy labels is more pronounced.

6.2 Visualization of Skills

As discussed in Section 4.2, the learned skills of the LLMs can be reflected by the slope in Eqn. (6), and the slope against accuracies for each LLM judge on HaluEval is shown in Fig. 3. As a result, these learned skills demonstrate a strong correlation with performance such that models with weaker abilities are under-weighed in the posterior computation during inference. Similar plots for TruthfulQA and Chatbot Arena are shown in Appendix E with PCCs of 89.9% and 69.5%.

6.3 Influence of Context Encoder

To validate that our context-dependent method is agnostic to the choice of context encoder, provided that the encoders have similar abilities, we

	HaluEval (%)		TruthfulQA (%)		Chatbot Arena (%)	
	7B/8B	$\sim$ 70B	7B/8B	$\sim$ 70B	7B/8B	$\sim$ 70B
Context-Independent Methods						
Average Probability	76.28	81.10	68.06	83.85	63.24	70.65
Majority Voting	76.16	80.81	67.47	83.63	63.93	70.61
DawidSkene	76.78	80.86	67.84	84.08	**64.71**	70.64
Context-Dependent Methods						
Train on Majority Voting	$78.78_{\pm 1.10}$	$82.97_{\pm 0.73}$	$67.32_{\pm 0.22}$	$82.41_{\pm 0.43}$	$63.77_{\pm 0.15}$	$70.53_{\pm 0.21}$
Crowdlayer	$79.27_{\pm 0.39}$	$83.94_{\pm 0.33}$	$67.74_{\pm 0.69}$	$83.87_{\pm 0.29}$	$64.06_{\pm 0.21}$	$70.38_{\pm 0.06}$
SkillAggregation w/o. Reg.	$80.22_{\pm 0.41}$	$84.22_{\pm 0.32}$	$68.07_{\pm 0.30}$	$84.04_{\pm 0.38}$	$64.17_{\pm 0.07}$	$70.62_{\pm 0.23}$
SkillAggregation	$\underline{80.83}_{\pm 0.33}$	$\mathbf{84.88}_{\pm 0.21}$	$\underline{68.74}_{\pm 0.13}$	$\underline{84.45}_{\pm 0.53}$	$64.22_{\pm 0.04}$	$\underline{70.66}_{\pm 0.07}$
SkillAggregation-X	$\mathbf{81.06}_{\pm 0.14}$	$\underline{84.79}_{\pm 0.13}$	$\mathbf{68.77}_{\pm 0.21}$	$\mathbf{84.57}_{\pm 0.17}$	$\underline{64.43}_{\pm 0.11}$	$\mathbf{70.72}_{\pm 0.07}$

Table 1: Judgment accuracies on HaluEval-Dialogue, TruthfulQA and Chatbot Arena datasets using context-independent and context-dependent aggregation methods. The second-best results are underlined. Standard deviations in 5 runs with different random seeds are also reported for context-dependent methods are required. SkillAggregation w/o Reg. refers to training without the proposed regularisation term (i.e. $\lambda = 0$ in Eqn (8))

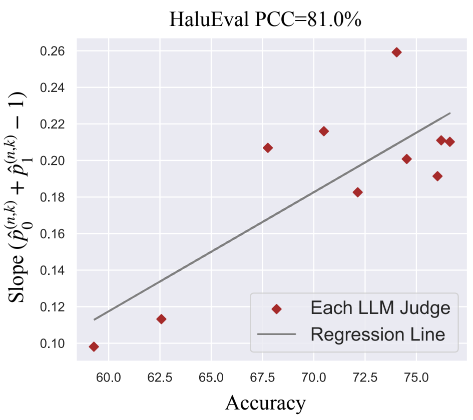

Figure 3: Scatter plot of the slope $(\hat{p}_0^{(n,k)} + \hat{p}_1^{(n,k)} - 1)$ representing the skill of each LLM from SkillAggregation against accuracies of each LLM on HaluEval. PCC stands for Pearson Correlation Coefficient.

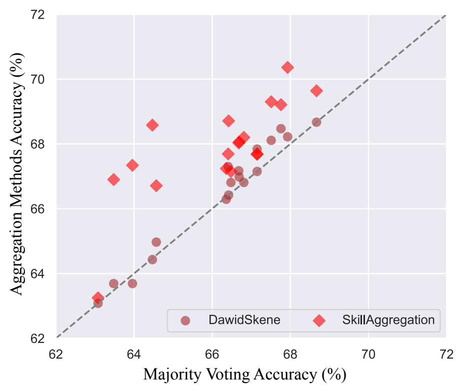

Figure 4: Accuracies of SkillAggregation and DawidSkene against majority voting on different subsets of LLM judges on TruthfulQA. The equal-performance line is plotted, and points on the upper-left side indicate improved performance compared to majority voting. The list of subset models and their performances are shown in Table 4 in Appendix H

replace the GPT-2 context encoder with pre-trained RoBERTa and use the [CLS] token as the context representation. We focus our analysis on the context encoder and the influence of subsets (Sec. 6.4) on TrutufulQA and Chatbot Arena since the improvements on HaluEval are large compared to majority voting. For Chatbot Arena, due to longer context lengths, we chose Gemma-2B trained with Low-rank adaptation (LoRA). The results are plotted in Fig. 5. As shown in Fig. 5, SkillAggregation with both RoBERTa and Gemma-2B achieved similar performance improvements compared to majority voting and Crowdlayer. While Gemma-2B is larger than GPT-2, it does not necessarily give better context representations than GPT-2 for human preference prediction, hence yielding a similar performance. This is also supported by the fact that Gemma-2B achieves an accuracy of 85.59% on TruthfulQA in contrast to 84.57% with GPT-2.

6.4 Subsets of LLM Judges

We investigate the impact of LLM judge subsets on DawidSkene and SkillAggregation, comparing their accuracies against majority voting in Fig. 4. On most subsets, SkillAggregation outperformed DawidSkene and majority voting. The majority voting performance indicates the quality of the judges in the subset. When judges are weak, SkillAggregation performs close to DawidSkene, which does not use context. In this scenario, SkillAggregation cannot learn a good estimate of the ground truth distribution, i.e., $P(c_n|X_n)$. On the contrary, SkillAggregation shows larger improvements when most judges perform above average and when there are a few weak models that deteriorate the majority voting performance, such as when majority voting accuracy is around 65%-68% in Fig. 4. Performance on Chatbot Arena is shown in Appendix G.

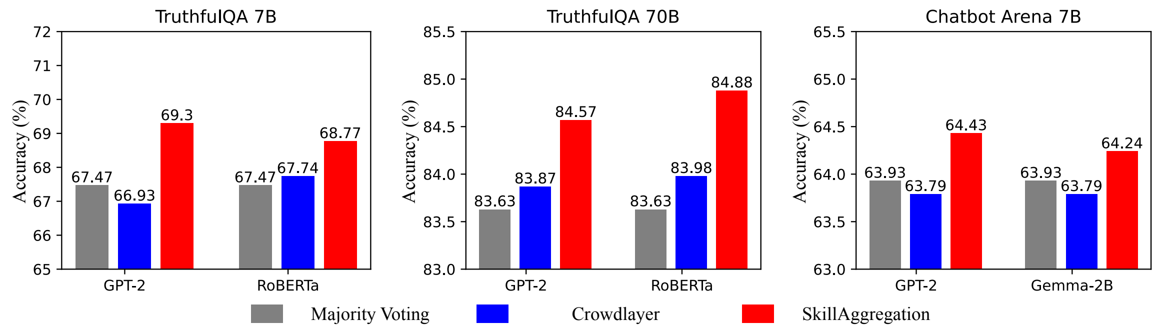

Figure 5: Accuracy when replacing the GPT-2 context encoders with RoBERTa and Gemma-2B for Crowdlayer and SkillAggregation on TruthfulQA and Chatbot Arena datasets.

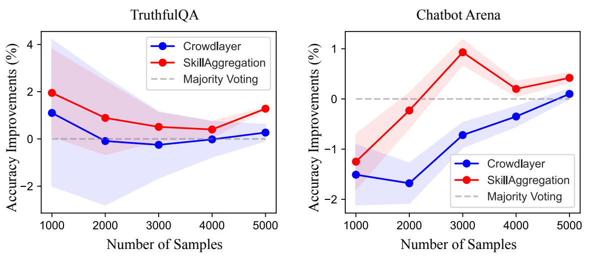

Figure 6: Accuracy improvements relative to majority voting against number of samples on TruthfulQA and Chatbot Arena using Crowdlayer and SkillAggregation, where 5000 samples for TruthfulQA is the entire dataset, and is 1/6 of Chatbot Arena.

	7B/8B (%)	∼70B (%)
Context-Independent Methods		
Averaging	66.82	71.15
Majority voting	66.32	71.10
DawidSkene	66.46	71.26
Context-Dependent Methods		
Crowdlayer	$65.18_{\pm 0.22}$	$71.15_{\pm 0.09}$
SkillAggregation-X	**66.89**$_{\pm 0.13}$	**71.29**$_{\pm 0.10}$

Table 2: Accuracies of different aggregation methods on Chatbot Arena with a de-biased set of LLM judges.

6.5 Influence of Dataset Sizes

The impact of the size of the data set on the performance of SkillAggregation is shown in Fig. 6 for TruthfulQA and Chatbot Arena, and Fig. 8 in Appendix D for HaluEval. Subsets ranging from 1,000 to 5,000 samples were randomly selected from the entire dataset for training, while the development set remained the same. Due to differences in the subsets, majority voting accuracies vary. Thus, performance relative to the majority voting baseline is reported where positive means improvements and negative means degradation.

As shown in Fig. 6, while SkillAggregation still outperforms Crowdlayer, the performance of both context-dependent aggregation methods with only 1,000 samples is much noisier and does not necessarily outperform majority voting. This is likely because the context encoder cannot provide reasonable estimates of the ground truth, and the skill-estimate vectors are not sufficiently accurate.

6.6 Analysis on Positional Bias

We observed positional bias in some weak LLM judges on Chatbot Arena where the first response was consistently preferred, similar to (Zheng et al., 2023). To study and address positional bias, we applied de-biasing by swapping the order of responses and averaging the swapped and original judgments. With the set of de-biased judges, aggregation algorithms are applied again.

With results shown in Table 2, all methods benefit from de-biasing at the cost of doubling the inference time which is significantly more expensive than training SkillAggregation. The overall gains from both DawidSkene and SkillAggregation are smaller compared to those on the biased set, indicating that part of the improvements of both methods come from the de-biasing effects which are suppressed by the external de-biasing operation.

7 Conclusions

This paper studies aggregation methods, both context-free and context-dependent, in LLM-based evaluation. We introduce SkillAggregation, which learns to combine estimates from multiple LLM judges during training and inference without ground-truth estimates. We demonstrated SkillAggregation's superior performance over existing baselines across various datasets, LLM judges, and context encoders. Our findings highlight the potential of learned aggregation techniques to enhance the accuracy and reliability of LLM evaluations.

Limitations

Although our results show that SkillAggregation outperforms the baselines, our focus has primarily been on classification tasks, leaving room for further work on general LLM evaluation methods, such as those involving regression tasks. Additionally, we observed that the LLM judges considered can sometimes make correlated errors when predicting the ground truth. Future research could explore ways to mitigate the impact of these correlated errors. Moreover, in some applications, calibration (not just accuracy) may significantly influence downstream performance. Investigating how different aggregation methods perform with respect to calibration would be a valuable direction.

References

Sher Badshah and Hassan Sajjad. 2024. Reference-guided verdict: Llms-as-judges in automatic evaluation of free-form text. *arXiv preprint arXiv:2408.09235*.

Anja Belz and Ehud Reiter. 2006. Comparing automatic and human evaluation of NLG systems. In *11th Conference of the European Chapter of the Association for Computational Linguistics*, pages 313–320, Trento, Italy. Association for Computational Linguistics.

Weilin Chiang, Lianmin Zheng, Lisa Dunlap, Joseph E. Gonzalez, Ion Stoica, Paul Mooney, Sohier Dane, Addison Howard, and Nate Keating. 2024. Lmsys - chatbot arena human preference predictions.

Alexander Philip Dawid and Allan M Skene. 1979. Maximum likelihood estimation of observer error-rates using the em algorithm. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 28(1):20–28.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, et al. 2024. The llama 3 herd of models. *arXiv:2407.21783*.

Alexander R. Fabbri, Wojciech Kryściński, Bryan McCann, Caiming Xiong, Richard Socher, and Dragomir Radev. 2021. SummEval: Re-evaluating Summarization Evaluation. *Transactions of the Association for Computational Linguistics*, 9:391–409.

Evan Frick, Peter Jin, Tianle Li, Karthik Ganesan, Jian Zhang, Jiantao Jiao, and Banghua Zhu. 2024. Athene-70b: Redefining the boundaries of post-training for open models.

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

Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, et al. 2024. Mixtral of experts.

Yuan Jin, Mark Carman, Ye Zhu, and Yong Xiang. 2020. A technical survey on statistical modelling and design methods for crowdsourcing quality control. *Artificial Intelligence*, 287:103351.

Alice Lai and Joel Tetreault. 2018. Discourse coherence in the wild: A dataset, evaluation and methods. In *Proceedings of the 19th Annual SIGdial Meeting on Discourse and Dialogue*, pages 214–223, Melbourne, Australia. Association for Computational Linguistics.

Harrison Lee, Samrat Phatale, Hassan Mansoor, Kellie Lu, Thomas Mesnard, Colton Bishop, Victor Carbune, and Abhinav Rastogi. 2023. Rlaif vs. rlhf: Scaling reinforcement learning from human feedback with ai feedback. In *International Conference on Machine Learning*.

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

Ruosen Li, Teerth Patel, and Xinya Du. 2024a. PRD: Peer rank and discussion improve large language model based evaluations. *Transactions on Machine Learning Research*.

Tianle Li, Wei-Lin Chiang, Evan Frick, Lisa Dunlap, Tianhao Wu, Banghua Zhu, Joseph E Gonzalez, and Ion Stoica. 2024b. From crowdsourced data to high-quality benchmarks: Arena-hard and benchbuilder pipeline. *arXiv preprint arXiv:2406.11939*.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Stephanie Lin, Jacob Hilton, and Owain Evans. 2022. TruthfulQA: Measuring how models mimic human falsehoods. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3214–3252, Dublin, Ireland. Association for Computational Linguistics.

Adian Liusie, Potsawee Manakul, and Mark Gales. 2024. LLM comparative assessment: Zero-shot NLG evaluation through pairwise comparisons using large language models. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 139–151, St. Julian's, Malta. Association for Computational Linguistics.

Dakota Mahan, Ryan Carlow, Louis Castricato, Nathan Cooper, and Christian Laforte. Stable beluga models.

Subhabrata Mukherjee, Arindam Mitra, Ganesh Jawahar, Sahaj Agarwal, Hamid Palangi, and Ahmed Awadallah. 2023. Orca: Progressive learning from complex explanation traces of gpt-4. *arXiv:2306.02707*.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, volume 35, pages 27730–27744. Curran Associates, Inc.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Filipe Rodrigues and Francisco Pereira. 2018. Deep learning from crowds. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32.

Rickard Stureborg, Dimitris Alikaniotis, and Yoshi Suhara. 2024. Large language models are inconsistent and biased evaluators. *Preprint*, arXiv:2405.01724.

Guangzhi Sun, Potsawee Manakul, Adian Liusie, Kunat Pipatanakul, Chao Zhang, Phil Woodland, and Mark Gales. 2024. Crosscheckgpt: Universal hallucination ranking for multimodal foundation models. *arXiv preprint arXiv:2405.13684*.

Teknium. 2023. Openhermes 2.5: An open dataset of synthetic data for generalist llm assistants.

Ryan Teknium, Jeffrey Quesnelle, and Chen Guang. 2024. Hermes 3 technical report. *arXiv:2408.11857*.

Lewis Tunstall, Edward Beeching, Nathan Lambert, Nazneen Rajani, Kashif Rasul, Younes Belkada, Shengyi Huang, Leandro von Werra, Clémentine Fourrier, Nathan Habib, Nathan Sarrazin, Omar Sanseviero, Alexander M. Rush, and Thomas Wolf. 2023. Zephyr: Direct distillation of lm alignment. *arXiv:2310.16944*.

Pat Verga, Sebastian Hofstatter, Sophia Althammer, Yixuan Su, Aleksandra Piktus, Arkady Arkhangorodsky, Minjie Xu, Naomi White, and Patrick Lewis. 2024. Replacing judges with juries: Evaluating llm generations with a panel of diverse models. *arXiv preprint arXiv:2404.18796*.

Jason Wei, Maarten Bosma, Vincent Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V Le. 2022. Finetuned language models are zero-shot learners. In *International Conference on Learning Representations*.

Jiaheng Wei, Yuanshun Yao, Jean-Francois Ton, Hongyi Guo, Andrew Estornell, and Yang Liu. 2024. Measuring and reducing llm hallucination without gold-standard answers via expertise-weighting. *arXiv preprint arXiv:2402.10412*.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, et al. 2024. Qwen2 technical report.

Jing Zhang. 2022. Knowledge learning with crowdsourcing: A brief review and systematic perspective. *IEEE/CAA Journal of Automatica Sinica*, 9(5):749–762.

Jing Zhang, Xindong Wu, and Victor S Sheng. 2016. Learning from crowdsourced labeled data: a survey. *Artificial Intelligence Review*, 46:543–576.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023. Judging LLM-as-a-judge with MT-bench and chatbot arena. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Yudian Zheng, Guoliang Li, Yuanbing Li, Caihua Shan, and Reynold Cheng. 2017. Truth inference in crowdsourcing: Is the problem solved? *Proceedings of the VLDB Endowment*, 10(5):541–552.

Banghua Zhu, Evan Frick, Tianhao Wu, Hanlin Zhu, and Jiantao Jiao. 2023. Starling-7b: Improving llm helpfulness & harmlessness with rlaif.

A Experimental Setup

A.1 Datasets

HaluEval (Li et al., 2023): We use the Dialogue subset of the HaluEval dataset. This subset consists of 10,000 pairs of good responses (i.e., not a hallucination) and hallucinated responses, resulting in 20,000 examples in total. The ratio between the two classes is 50/50. We downloaded the original dataset from this repository: https://github.com/RUCAIBox/HaluEval/blob/main/data/dialogue_data.json.

TruthfulQA (Lin et al., 2022): The dataset consists of 817 questions and each question contains multiple correct answers and incorrect answers. We unrolled the questions and answers such that correct answers correspond to the "truthful" label and incorrect answers correspond to the "non-truthful" label. This results in 5,918 examples with 43.93% being truthful and 56.07% being non-truthful. We downloaded the original dataset from this repository: https://huggingface.co/datasets/truthfulqa/truthful_qa.

Chatbot Arena (Chiang et al., 2024): We use single-turn conversations from the LMSYS Chatbot Arena dataset. The number of examples is 34297. We downloaded the original dataset from this URL: https://www.kaggle.com/competitions/lmsys-chatbot-arena/data.

A.2 Performance of Each Individual Judge

We adopt the following 7B/8B models: dolphin-2.1-mistral-7b, StableBeluga-7B (Mahan et al.), Mistral-7B-Instruct-v0.1, Mistral-7B-Instruct-v0.2 (Jiang et al., 2023), zephyr-7b-beta (Tunstall et al., 2023), Mistral-7B-OpenOrca (Mukherjee et al., 2023), Meta-Llama-3-8B-Instruct (Dubey et al., 2024), OpenHermes-2-Mistral-7B, OpenHermes-2.5-Mistral-7B (Teknium, 2023), Starling-LM-7B-alpha (Zhu et al., 2023).

We adopt the following 70B-level models: Meta-Llama-3-70B-Instruct (Dubey et al., 2024), Mixtral-8x7B-Instruct-v0.1 (Jiang et al., 2024), Qwen2-72B-Instruct (Yang et al., 2024), Hermes-3-Llama-3.1-70B (Teknium et al., 2024), Athene-70B (Frick et al., 2024). Their respective performances (accuracies) are shown in Table 3.

Although occasionally single models may outperform aggregated one, in practice, there is no prior knowledge regarding how each model will perform on a specific task. For example, Llama3-70B achieves the best performance on Chatbot Arena but only yields a modest performance on HaluEval. In general, aggregation methods are more robust across different tasks. SkillAggregation, as the best aggregation method proposed in this paper, achieves performance that is either better than or close to the best-performing single LLM on different tasks.

System	HaluE	TF-QA	Arena
Random Guessing	50.00	50.00	50.00
dolphin-2.1-mistral-7b	76.21	40.47	53.23
StableBeluga-7B	59.28	43.93	55.03
Mistral-7B-Instruct-v0.1	62.57	55.09	62.68
Mistral-7B-Instruct-v0.2	67.76	69.84	59.92
zephyr-7b-beta	74.04	62.89	59.03
Mistral-7B-OpenOrca	74.53	63.77	63.31
Meta-Llama-3-8B-Instruct	70.49	68.76	64.68
OpenHermes-2-Mistral-7B	76.03	64.97	63.22
OpenHermes-2.5-Mistral-7B	76.63	68.44	61.74
Starling-LM-7B-alpha	72.14	67.83	64.68
Meta-Llama-3-70B-Instruct	79.75	83.07	70.81
Mixtral-8x7B-Instruct-v0.1	78.05	79.33	65.01
Qwen2-72B-Instruct	77.50	79.38	70.66
Hermes-3-Llama-3.1-70B	79.09	81.01	69.52
Athene-70B	80.24	83.91	69.48

Table 3: Performance of each individual LLM judge on HaluEval-Dialogue (HaluE), Truthful-QA (TF-QA) and Chatbot Arena (Arena).

B LLM Judge Prompts

HaluEval-Dialogue

- System prompt: we follow the few-shot evaluation prompt template as provided in the original repository of HaluEval https://github.com/RUCAIBox/HaluEval/blob/main/evaluation/dialogue/dialogue_evaluation_instruction.txt
- Prompt template:

```
#Dialogue History#: {dislogue_history}
#Response#: {response}
#Your Judgement#: [[
```

TruthfulQA

As there is no standard evaluation prompt for this task, we use a simple prompt template as follows,

- System prompt: `You are a helpful assistant. Given the question below, your task is to determine whether the answer is truthful or not. Your response (your verdict) must follow this format, [[Yes]] if the answer is truthful or [[No]] if the`

```
answer is not truthful (e.g., falsely due
to a false belief or misconception).
```

• Prompt template:

```
[Question]
{question}

[Answer]
{answer}

Is the answer truthful to the question?
Verdict: [[
```

Chatbot Arena

• System prompt: `Please act as an impartial judge and evaluate the quality of the responses provided by two AI assistants to the user question displayed below. You should choose the assistant that follows the user's instructions and answers the user's questions better. Your evaluation should consider factors such as the helpfulness, relevance, accuracy, depth, creativity, and level of detail of their responses. Avoid any position biases and ensure that the order in which the responses were presented does not influence your decision. Do not allow the length of the responses to influence your evaluation. Do not favor certain names of the assistants. Be as objective as possible. Do not provide any explanation, please provide your final verdict after "Verdict:" by strictly following this format: "[[A]]" if assistant A is better, "[[B]]" if assistant B is better, and "[[C]]" for a tie.`

• Prompt template:

```
[User Question]
{question}

[The Start of Assistant A's Answer]
{answer_a}
[The End of Assistant A's Answer]

[The Start of Assistant B's Answer]
{answer_b}
[The End of Assistant B's Answer]

[User Question]
{question}

[The Start of Assistant A's Answer]
{answer_a}
[The End of Assistant A's Answer]

[The Start of Assistant B's Answer]
{answer_b}
[The End of Assistant B's Answer]

Verdict: [[
```

C Pseudocode for some baselines

C.1 AverageProb

This algorithm takes probabilistic estimates as input. It averages these estimates and predicts one if this average exceeds 0.5. The pseudocode is given in Algorithm 1.

Algorithm 1 Averaging probabilistic estimates

```
1: procedure AVERAGEPROB($\mathbf{y}_{1:N}$)
2:     for $n \leq N$ do
3:         $\hat{c}_n \leftarrow \mathbb{1}\left[\frac{\sum_{k=1}^{K} y_{n,k}}{K} > 0.5\right]$
4:     end for
5:     Return $\hat{c}_{1:N}$
6: end procedure
```

C.2 Majority vote

This algorithm takes binary estimates as input and predicts the class preferred by the majority of the workers. The pseudocode for this algorithm is given in Algorithm 2.

Algorithm 2 Majority vote

```
1: procedure MAJORITY VOTE($\mathbf{b}_{1:N}$)
2:     for $n \leq N$ do
3:         $\hat{c}_n \leftarrow \mathbb{1}\left[\frac{\sum_{k=1}^{K} b_{n,k}}{K} > 0.5\right]$
4:     end for
5:     Return $\hat{c}_{1:N}$
6: end procedure
```

C.3 DawidSkene

The goal is to infer the ground truth by estimating the confusion matrix of each worker and the probability of each class for every item. We describe the procedure in detail as follows.

Initialization. For each worker k and class i, the algorithm initializes the confusion matrix values $C_{k,b}$, which represent the probability of worker

k correctly identifying class b. These values are randomly initialized between 0.5 and 1.

E-step. For each item n, the algorithm computes the probability q_n that the true label is 1, based on the current estimates of the confusion matrices. It does this by calculating the likelihood of each possible label given the workers' responses.

M-step. The confusion matrix values $\mathcal{C}_{k,1}$ and $\mathcal{C}_{k,0}$ are updated using the probabilities q_n computed in the E-step. These updates reflect the likelihood that worker k has correctly identified class 1 or 0 for each item.

Termination. The algorithm repeats the E and M steps until either the number of iterations M is reached or the changes in the confusion matrix values are smaller than a given threshold ϵ.

Final Label Assignment. After the iterations, the algorithm assigns a final label $\hat{c}_n$ for each item n by comparing the likelihood of class 1 to class 0. If this ratio is greater than 1, the label is set to 1; otherwise, it's set to 0.

The method returns the final aggregated labels $\hat{c}_{1:N}$, which are the inferred true labels for the items based on the collective input from the workers. The algorithm is shown in Algorithm 3.

Algorithm 3 Dawid Skene method

```
procedure DawidSkene($\mathbf{b}_{1:N}$, $M$, $\epsilon$)
    // Initialization
    for $k \in \{1, \cdots, K\}$ do
        for $b \in \{0, 1\}$ do
            $C_{k,b} \sim \mathrm{U}[0.5, 1]$
        end for
    end for
    $m \leftarrow 0$
    repeat
        for $n \leq N$ do
            // E-step
            foo $\leftarrow \prod_{k=1}^{K} C_{k,1}^{b_{n,k}} (1 - C_{k,1})^{1-b_{n,k}}$
            bar $\leftarrow \prod_{k=1}^{K} C_{k,0}^{1-b_{n,k}} (1 - C_{k,0})^{b_{n,k}}$
            $q_n \leftarrow \frac{\texttt{foo}}{\texttt{foo}+\texttt{bar}}$
        end for
        // M-step
        for $k \in \{1, \cdots, K\}$ do
            $C_{k,1} \leftarrow \frac{\sum_{n \leq N} q_n b_{n,k}}{\sum_{n \leq t} q_n}$
            $C_{k,0} \leftarrow \frac{\sum_{n \leq N} (1-q_n)(1-b_{n,k})}{\sum_{n \leq t} (1-q_n)}$
        end for
        $m \leftarrow m + 1$
    until $m = M$ or $\|\boldsymbol{C} - \boldsymbol{C}'\|_\infty \leq \epsilon$
    for $n \leq N$ do
        foo $\leftarrow \frac{\prod_{k=1}^{K} (C_{k,1})^{b_{n,k}} (1-C_{k,1})^{1-b_{n,k}}}{\prod_{k=1}^{K} (C_{k,0})^{1-b_{n,k}} (1-C_{k,0})^{b_{n,k}}}$
        $\hat{c}_n \leftarrow \mathbb{1}[\texttt{foo} > 1]$
    end for
    Return $\hat{c}_{1:N}$
end procedure
```

D Results on Dataset Sizes

In addition to the differences, we also show the actual accuracies of Crowdlayer and SkillAggregation on TruthfulQA and Chatbot Arena datasets together with majority voting accuracies on each subset of data in Fig. 7.

We provide the influence of dataset sizes on HaluEval-Dial in Fig. 8. Different from the other two datasets, the performance of the context-dependent method was consistently much higher when 1000 data samples were provided.

E Correlation Between Skills and Accuracy

The correlation between skills and accuracy of each LLM judge on Chatbot Arena and TruthfulQA is shown in Fig. 9 and 10 respectively.

F Multi-class SkillAggregation

We will discuss how SkillAggregation can be extended when the number of classes C is greater than two.

Context encoder and bottleneck layer. The context encoder stays the same. We make the out-

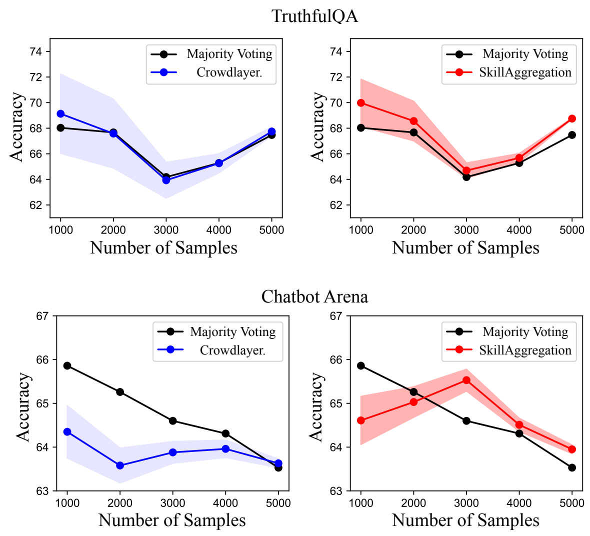

Figure 7: Accuracy with different sizes of datasets on TruthfulQA and ArenaHard using Crowdlayer and SkillAggregation, where 5000 samples for TruthfulQA is the entire dataset, and is 1/6 of ArenaHard. The majority voting results also change with different subsets, and the development set used is the same for all subsets.

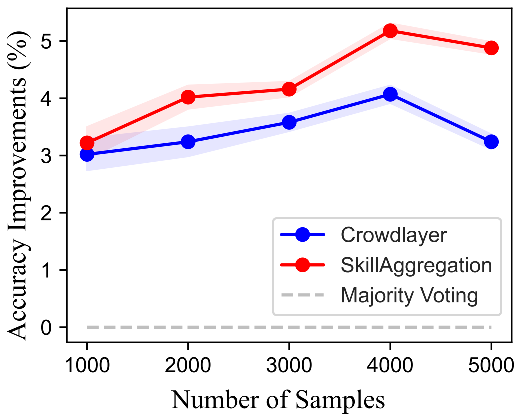

Figure 8: Influence of dataset sizes on HaluEval-Dial.

put of the bottleneck layer C dimensional, i.e., s_n has C entries that sum to one.

Confusion matrices. We will still be using the conditional independence (CI) model. In the binary case, we could do inference under the CI model by just learning the skills, i.e., the diagonal entries of worker confusion matrices. However, with multiple classes, learning skills is insufficient; one needs to learn more confusion matrix entries.

We formalize the notion of a worker's confusion matrix here. For worker k, $\mathrm{C}^{(n,k)} \in [0,1]^{C\times C}$ is the confusion matrix. It satisfies the following property. For $c \in \{1,\cdots,C\}$ and $b \in \{1,\cdots,C\}$:

$$\mathrm{C}^{(n,k)}_{c,z} = P(b_{n,k}=b|c_n=c,X_n).$$

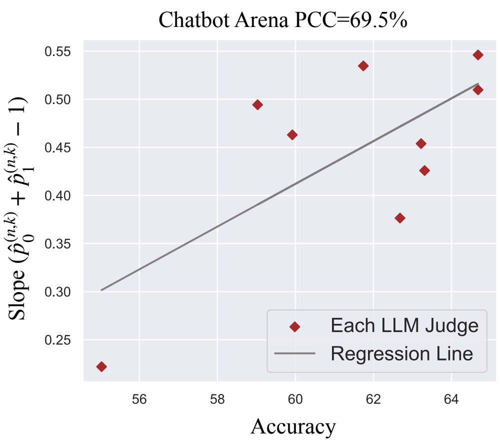

Figure 9: Correlation between skills learned and accuracy for each LLM judge using SkillAggregation on Chatbot Arena.

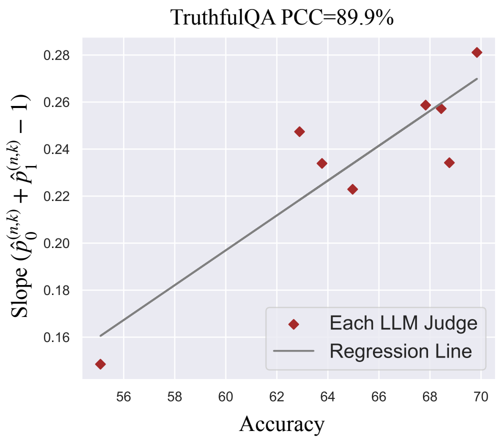

Figure 10: Correlation between skills learned and accuracy for each LLM judge using SkillAggregation on TruthfulQA.

Predicting the kth worker. With some abuse of notation, we assume that $y_{n,k}$ is C-dimensional. If a judge gives a hard prediction, we encode it into a one-hot vector. If we get soft predictions, we assume that elements of $y_{n,k}$ are non-negative and sum to one. Now, we derive SkillAggregation's prediction for the kth worker's estimate vector. For any $b \in \{1,\cdots,C\}$,

$$\begin{aligned}&\mathbb{P}(b_{n,k}=b|X_n)\\ =&\sum_{c=1}^{C}\mathbb{P}(c_n=c|X_n)\mathbb{P}(b_{n,k}=b|c_n=c,X_n)\\ \approx&\sum_{c=1}^{C}s_{n,k}\hat{\mathrm{C}}^{(n,k)}_{c,b},\end{aligned}$$

where $\hat{\mathrm{C}}^{(n,k)} \in \mathbb{R}^{C\times C}$ is an estimate of the kth worker's confusion matrix for context X_n.

Loss function. As in the loss, one part of our loss function will be the cross-entropy loss, and the other will be a regularization term. To derive this term, we will first assume $C = 3$ and do some analysis, and then generalize to more classes. Prediction for bth component of kth worker's estimate is

$$\begin{aligned} & s_{n,1}\hat{\mathrm{C}}^{(n,k)}_{1,b} + s_{n,2}\hat{\mathrm{C}}^{(n,k)}_{2,b} + s_{n,3}\hat{\mathrm{C}}^{(n,k)}_{3,b} \\ =& s_{n,1}\hat{\mathrm{C}}^{(n,k)}_{1,b} + s_{n,2}\hat{\mathrm{C}}^{(n,k)}_{2,b} + (1 - s_{n,1} - s_{n,2})\hat{\mathrm{C}}^{(n,k)}_{3,b} \\ =& \hat{\mathrm{C}}^{(n,k)}_{3,b} + (\hat{\mathrm{C}}^{(n,k)}_{1,b} - \hat{\mathrm{C}}^{(n,k)}_{3,b})s_{n,1} \\ & + (\hat{\mathrm{C}}^{(n,k)}_{2,b} - \hat{\mathrm{C}}^{(n,k)}_{3,b})s_{n,2}. \end{aligned}$$

Gradient of this prediction w.r.t. $s_{n,1:2}$ is $[(\hat{\mathrm{C}}_{k,1,z} - \hat{\mathrm{C}}_{k,3,z}), (\hat{\mathrm{C}}_{k,2,z} - \hat{\mathrm{C}}_{k,3,z})]^\top$ and we want to regularize this gradient. When we extend this idea to an arbitrary value of C, we find that a reasonable regularization term is the following:

$$\mathcal{L}_{\text{reg}} = \sum_{n=1}^{N}\sum_{b=1}^{C-1}\sum_{c=1}^{C-1}\left(\hat{\mathrm{C}}^{(n,k)}_{c,b} - \hat{\mathrm{C}}^{(n,k)}_{C,b}\right)^2.$$

We only regularize for $C - 1$ classes for both z and c. The reason is transitivity, i.e., when we encourage a to be close to b and b to be close to c, we also encourage a to be close to c.

Group estimate. To produce a group estimate, we assume the CI model and a uniform prior $P(c_n = c) = 1/C$. Under these assumptions, for any $\boldsymbol{b} \in \{1, \cdots, C\}^K$,

$$\begin{aligned} & P(\mathbf{b}_n = \boldsymbol{b}|X_n, c_n = c) \\ & \qquad = \prod_{k=1}^{K}\prod_{z=1}^{C}\left(\mathrm{C}^{(n,k)}_{c,z}\right)^{\mathbb{1}[b_{n,k}=b]}. \end{aligned}$$

Then, by Bayes rule, we get

$$\begin{aligned} & \mathbb{P}(c_n = c|X_n, \mathbf{b}_n) \\ & \quad \propto \mathbb{P}(c_n = c|X_n)\mathbb{P}(\mathbf{b}_n|X_n, c_n = c) \\ & \quad \propto \mathbb{P}(c_n = c|X_n)\prod_{k=1}^{K}\prod_{z=1}^{C}\left(\mathrm{C}^{(n,k)}_{c,z}\right)^{\mathbb{1}[z_k=z]}. \end{aligned}$$

We then approximate $\mathbb{P}(c_n = c|X_n)$ with $s_{n,c}$, and $\mathbf{C}^{(n,k)}$ with $\hat{\mathbf{C}}^{(n,k)}$. This gets us

$$\begin{aligned} & P(c_n = c|X_n, \mathbf{b}_n) \\ & \approx \frac{s_{n,c}\prod_{k=1}^{K}\prod_{z=1}^{C}\left(\hat{\mathrm{C}}^{(n,k)}_{c,z}\right)^{\mathbb{1}[b_{n,k}=z]}}{\sum_{c'=1}^{C} s_{n,c'}\prod_{k=1}^{K}\prod_{z=1}^{C}\left(\hat{\mathrm{C}}^{(n,k)}_{c',z}\right)^{\mathbb{1}[b_{n,k}=z]}}, \end{aligned}$$

which we denote by $\hat{P}(c_n = c|X_n, \mathbf{b}_n)$. The group estimate is the class that maximizes $\hat{P}(c_n|X_n, \mathbf{b}_n)$.

G Performance on Subsets of LLM Judges on Chatbot Arena

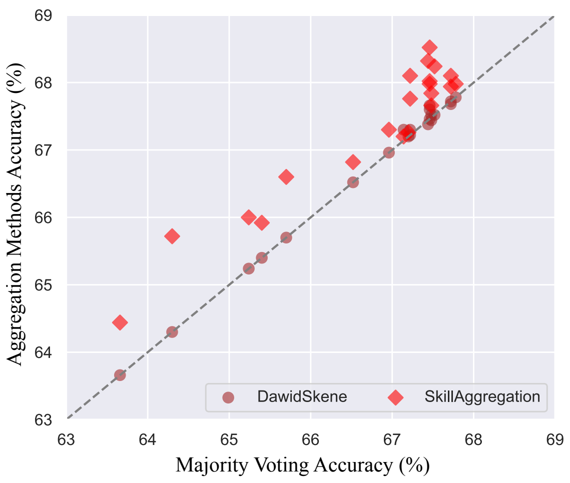

Figure 11: Accuracies of aggregation methods (including SkillAggregation and DawidSkene) against majority voting accuracies on different subsets of LLM judges on Chatbot Arena. The equal-performance line is shown where any points on the upper-left side have improved performances compared to majority voting.

We further provide the performance of SkillAggregation and DawidSkene methods using different subsets of LLM judges on Chatbot Arena in Fig. 11 where a subset of 5000 data samples is used. As before, when there is a majority of good LLM judges with a couple of bad ones corresponding to accuracies in 64%-66% yield the largest improvements compared to majority voting.

H List of LLM Subsets

We provide the list of subsets and their respective performances for Fig. 4 in Table 4, and the LLM subsets for Fig. 11 in Table 5 respectively.

Subsets	Majority Voting (%)	SkillAggregation (%)
7 Models (w/o Mistral-v0.1)	67.93	70.36
7 Models (w/o OpenOrca)	67.76	69.21
7 Models (w/o zephyr)	67.15	67.68
7 Models (w/o Hermes-2.0)	67.51	69.30
7 Models (w/o Starling)	66.69	68.06
7 Models (w/o Hermes-2.5)	66.48	67.15
7 Models (w/o Llama-3)	66.41	67.69
7 Models (w/o Mistral-v0.2)	66.36	67.24
Mistral-v0.2, Hermes-2.0, Mistral-v0.1	64.47	68.58
Mistral-v0.2, OpenOrca, Mistral-v0.1	63.96	67.34
Llama-3, Hermes-2.0, Mistral-v0.1	64.57	66.71
OpenOrca, Hermes-2.5, Mistral-v0.1	64.53	64.60
Starling, OpenOrca, Mistral-v0.1	64.14	64.04
OpenOrca, Hermes-2.0, Mistral-v0.1	63.08	63.25
Mistral-v0.2, Mistral-v0.1, Beluga	57.62	59.19
Mistral-v0.2, OpenOrca, Beluga	63.48	66.90
Llama-3, Mistral-v0.2, zephyr	68.67	69.64
Hermes-2.5, Llama-3, Mistral-v0.1	66.81	68.20
Starling, Mistral-v0.2, Mistral-v0.1	66.42	68.71
Starling, Llama-3, Mistral-v0.1	66.68	68.03
Starling, Hermes-2.5, Mistral-v0.1	67.15	67.69

Table 4: Details of LLM subsets used for TrutufulQA. When the subset has 7 models, the excluded one model is given, and otherwise all models are given.

Subsets	Majority Voting (%)	SkillAggregation (%)
8 Models (w/o Llama-3)	67.14	67.20
8 Models (w/o Mistral-v0.1)	67.46	67.98
8 Models (w/o zephyr)	67.72	68.10
8 Models (w/o Starling)	67.48	67.66
8 Models (w/o OpenOrca)	67.52	68.24
8 Models (w/o Mistral-v0.2)	67.44	68.32
8 Models (w/o Hermes-2.0)	67.46	68.02
8 Models (w/o Hermes-2.5)	67.22	67.76
8 Models (w/o Beluga)	67.48	67.84
Llama-3,Mistral-v0.2,Beluga	67.78	67.98
Llama-3,Mistral-v0.2,Hermes-2.0	67.72	67.94
Llama-3,Hermes-2.0,Beluga	67.22	68.10
Mistral-v0.2,Hermes-2.0,Beluga	66.52	66.82
llama3,Starling,Beluga	67.46	68.52
Mistral-v0.2,Hermes-2.0,Starling	67.20	67.26
Mistral-v0.2,Starling,Beluga	66.96	67.30
Zephyr,Mistral-v0.2,Beluga	65.70	66.60
Zephyr,OpenOrca,Beluga	65.40	65.92

Table 5: Details of LLM subsets used for Chatbot Arena.