Towards More Robust NLP System Evaluation: Handling Missing Scores in Benchmarks

Anonymous ACL submission

Abstract

The evaluation of natural language processing (NLP) systems is crucial for advancing the field, but current benchmarking approaches often assume that all systems have scores available for all tasks, which is not always practical. In reality, several factors such as the cost of running baseline, private systems, computational limitations, or incomplete data may prevent some systems from being evaluated on entire tasks. This paper formalize an existing problem in NLP research: benchmarking when some systems scores are missing on the task, and proposes a novel approach to address it. Our method utilizes a compatible partial ranking approach to impute missing data, which is then aggregated using the Borda count method. It includes two refinements designed specifically for scenarios where either task-level or instance-level scores are available. We also introduce an extended benchmark, which contains over 131 million scores, an order of magnitude larger than existing benchmarks. We validate our methods and demonstrate their effectiveness in addressing the challenge of missing system evaluation on an entire task. This work highlights the need for more comprehensive benchmarking approaches that can handle real-world scenarios where not all systems are evaluated on the entire task.

1 Introduction

Benchmarking and system evaluation are critical processes for assessing the performance of AI systems, providing a standardized means of comparing various models and techniques while keeping track of technological advancements (Ruder, 2021; Dehghani et al., 2021; Post, 2018). However, evaluating general-purpose systems, such as foundation models used for generative tasks (Lehman et al., 2023; Kocoń et al., 2023; OpenAI, 2023; Brown et al., 2020; Raffel et al., 2020), presents unique challenges. A single task, metric, or dataset may not be sufficient to effectively gauge their capabilities (Herbrich et al., 2006; Novikova et al., 2018; Sedoc and Ungar, 2020). Therefore, it is crucial to develop tools that can benchmark these systems on a multitude of tasks (Aribandi et al., 2021), enabling a comprehensive assessment of their overall performance (Peyrard et al., 2021).

In recent years, the field of NLP has made significant strides, with frequent emergence of new models (Lehman et al., 2023; Kocoń et al., 2023; Brown et al., 2020; OpenAI, 2023; Raffel et al., 2020; Liu et al., 2019; Fan et al., 2021) and techniques (Bommasani et al., 2021; Hupkes et al., 2022). To evaluate the performance of these systems across various tasks, datasets, and metrics (Colombo et al., 2022c) have been created. However, with the increasing complexity of these benchmarks, missing scores has become a significant challenge. Missing data can arise from a variety of sources, such as benchmarks that are too large or time-consuming to run (*e.g.*, BigBench has recently introduced MiniBench for these reasons (Srivastava et al., 2022)), high costs associated with reproducing experiments (*e.g.*, see Table 3 in (Artetxe et al., 2022)), incomplete datasets (see Table 5 in (Reid and Artetxe, 2022)), data collection errors, data cleaning procedures, data privacy concerns (particularly in-house datasets (Guibon et al., 2021)), and specialized expertise required to process niche datasets (Peng et al., 2019). In recent work, two main approaches have been followed to deal with missing scores, which are discarding data (Pfeiffer et al., 2022) or ignoring certain tasks (see Table 10 in (Lin et al., 2022) and Table 5 in (Martin et al., 2020)) or evaluations. However, these approaches are unsatisfactory as they can lead to biased and unreliable evaluations.

In this work, we address benchmarking NLP systems *when one or several systems cannot be evaluated on a specific task*. We propose the development of effective methods for aggregating

metrics that can handle missing data and enable a comprehensive assessment of system performance. Our approach will ensure the reliability and validity of NLP system evaluations and contribute to the creation of benchmarks that can be used to compare and evaluate NLP systems effectively. Specifically, our contributions are listed below.

1. **Introducing a new problem with a direct impact on NLP research:** benchmarking when there are missing system evaluations for an entire task, which has practical implications (Pfeiffer et al., 2022; Lin et al., 2022; Martin et al., 2020; Guibon et al., 2021; Peng et al., 2019).
2. **A novel method for benchmarking NLP systems with missing system scores.** We present a novel method that effectively tackles the issue of missing system evaluations for entire tasks. Our work includes a novel combinatorial approach for imputing missing data in partial rankings. It allows using standard rank aggregation algorithms such as Borda and offers two refinements tailored to the availability of either task-level or instance-level scores of the systems across different tasks.
3. **An extended benchmark for a comprehensive and accurate evaluation of NLP systems:** previous works on score aggregation relied on a benchmark of 250K scores (Colombo et al., 2022a; Peyrard et al., 2021), and did not release the system's input, output, and ground truth texts. In our work, we collected their scores and extended the benchmark by adding over 131M scores.
4. **Extensive validation of benchmarking methods:** Results show that our method effectively handles missing scores and is more robust than existing methods, affecting final conclusions.

2 Formulation & Related Work

2.1 General Considerations

Comparing systems with benchmarks. Benchmarking aims to determine the ranking of systems based on their scores to identify the best-performing systems. In this process, each system is evaluated on individual tests within a larger set and assigned a score according to a specific metric. Depending on the available information, two approaches are typically employed. When only **task-level** information is available (i.e., the system scores on each task), a **task-level aggregation** is utilized to obtain the final ranking. On the other hand, when **instance-level information** is available, i.e., the system scores on each instance of each task test set, an **instance-level aggregation** method is used to obtain the final system ranking. The mean aggregation has been adopted to aggregate at both the instance and task levels.

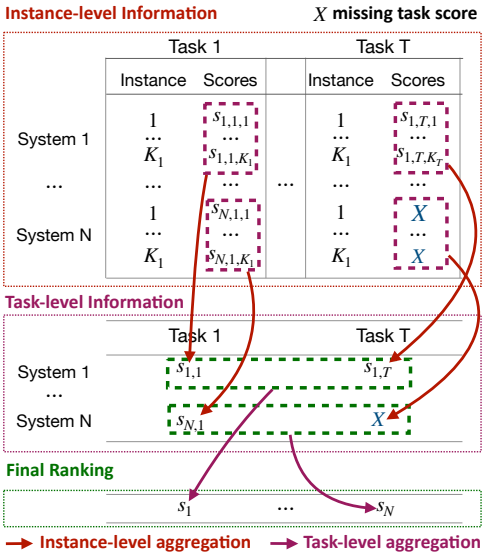


Figure 1: **Framework for benchmarking NLP systems with two information granularity**: *instance-level* (red above) and *task-level* (purple below). The final goal of benchmarking is to produce a ranking (green bottom). The instance-level aggregation allows for the derivation of task-level information, which is used to synthesize system performance via the final ranking (in green). **X** indicates the presence of missing values.

Benchmarking in the presence of missing data. As benchmarks and models continue to grow in size and complexity, the occurrence of missing system performance of entire tasks becomes increasingly common. This is particularly true in situations where one or more systems cannot be evaluated on a specific task due to factors such as the high cost of running the model or the extensive computational requirements of the benchmarks (Gehrmann et al., 2022a, 2021, 2022b). Fig. 1 illustrates the general framework (*i.e.*, with instance and system level).

2.2 Problem Formulation

This discussion will use notation similar to that in the previously mentioned work (Colombo et al., 2022a). In essence, we are dealing with a scenario where N systems are being compared based on their performance on T different tasks. Each task $t \in \{1, \dots, T\}$ has a specific metric m_t associated with it and has been evaluated on k test instances with $k \in \{1, \dots, K_t\}$, where K_t is the test size of task t. The score of each system on each instance of each test set is represented by

the real number $s_{n,t,k} \in \mathbb{R}$. The final goal of benchmarking is to output a ranking of each system according to some objective criterion. We denote by $\mathfrak{S}_N$ the symmetric group on N elements. With this objective in mind aggregating instance and task level information is equivalent to computing a permutation $\sigma \in \mathfrak{S}_N$ corresponding to the final ranking of the N systems. In this formalism, system i is the σ_i-th best system according to the considered aggregation. Equivalently, ordering $\pi = (\pi_1 \succ \pi_2 \succ \ldots \succ \pi_N)$ denotes that π_i is better than system π_{i+1} for all i. Let us first define the different granularity of benchmarking depending on whether we have access to instance scores.

Aggregating with Missing Task Level Information. *Given a set of scores* $(s_{n,t},\ 1 \leq n \leq N_t,\ 1 \leq t \leq T)$ *where* N_t *is the number of systems for which we have access to the score on task t, find a proper aggregation procedure.*

Thus the problem of task-level information aggregation boils down to finding f^T:

$$f^T : \underbrace{\mathcal{S}_{N_1} \times \cdots \times \mathcal{S}_{N_T}}_{T \text{ times}} \longrightarrow \mathfrak{S}_N. \qquad (1)$$

where $\mathcal{S}_{N_t} = (s_{n,t}, 1 \leq n \leq N_t)$ is the set of scores achieved by each system evaluated on the task t. Only N_t systems are evaluated on task t. In many cases, we not only have access to task-level performance but also individual instance-level scores. As a result, the challenge lies in effectively aggregating information at the instance level.

Aggregating Missing Instance Level Information. *Given a set of scores* $(s_{n,t,k},\ 1 \leq n \leq N_t,\ 1 \leq t \leq T,\ 1 \leq k \leq K_t)$ *where similarly as previously* N_t *is the number of systems for which we have access to the score on task t, find a proper aggregation procedure.*

Thus the problem of instance-level information aggregation boils down to finding f^I:

$$f^I : \underbrace{\mathcal{S}^1_{N_1} \times \cdots \times \mathcal{S}^{K_1}_{N_1} \times \cdots \times \mathcal{S}^k_{N_t} \times \cdots \times \mathcal{S}^1_{N_T} \times \cdots \mathcal{S}^{K_T}_{N_T}}_{T \sum_t K_t \text{ times}} \longrightarrow \mathfrak{S}_N. \qquad (2)$$

where $\mathcal{S}^k_{N_i} = (s_{n,t,k},\ 1 \leq n \leq N_i)$ is the set of the score achieved by each system evaluated on the task t for the specific instance k.

Remark 1. *In the context of complete ranking, which is also a classical setting for benchmarking NLP systems and has been addressed in (Colombo et al., 2022a), we have* $N_t = N$ *for all* $t \in [1, T]$.

2.3 Handling Complete Scores in NLP System Evaluation

The literature relies on two main techniques for aggregating score information to benchmark ML systems: mean aggregation and ranking-based aggregation.

Mean aggregation (σ^μ) is the default choice for practitioners. At the task level σ^μ is defined as :

$$\sigma^\mu = argsort\left(argsort\left[\frac{1}{T}\sum_{1\leq t\leq T} s_{n,t} \text{ for } 1 \leq n \leq N\right]\right)$$

and at the instance level :

$$\sigma^\mu = argsort\left(argsort\left[\frac{1}{T}\sum_{1\leq t\leq T}\frac{1}{K_t}\sum_{1\leq t\leq K_t} s_{n,t,k} \text{ for } 1 \leq n \leq N\right]\right)$$

where argsort(**u**) is the permutation that sorts the items in **u**. However, this approach has its limitations, particularly when evaluating tasks of different natures or using evaluation scores that are not on the same scale. Indeed in NLP, metrics can have different ranges (or even be unbounded) and systems are evaluated based on diverse criteria such as quality, speed, or number of parameters. In such cases, conventional rescaling or normalization techniques may not sufficiently capture the inherent difficulty of each task.

Ranking Based Aggregation To address the challenges mentioned, researchers have proposed ranking-based aggregations (Peyrard et al., 2021; Colombo et al., 2022a). These methods aggregate rankings instead of scores. In (Colombo et al., 2022a), the authors tackle the problem of generating a ranking by aggregating rankings, utilizing the Borda count method (see Ssec. E.1 for more details on Borda Count) known for its computational properties (Bartholdi et al., 1989; Dwork et al., 2001; Ali and Meilă, 2012). Extending the Borda count method is not a straightforward task either. Next, we present our aggregation procedure that can handle missing system scores on a whole task.

3 Ranking with missing system evaluation

In this section, we will outline our methodology for ranking multiple systems in multi-task benchmarks, even if some systems have not been evaluated on one or more tasks. We use the ranking and ordering notation interchangeably.

3.1 Partial Rankings

Mapping Scores to Partial Rankings To address the challenge of benchmarking with missing system evaluations, we propose a ranking-based approach that focuses on aggregating rankings rather

than directly combining scores. Suppose we have a specific task t with a task-level score denoted as S_{N_t}, or in the case of instance-level information, a task t and instance k with score $S^k_{N_t}$. In scenarios where there are missing evaluations at the task-level or instance-level, a *partial ranking* of systems is generated. A partial ordering represents an incomplete ranking that includes only a subset of items from a larger set. We denote the partial ordering of systems as $\pi^{N_t} = (\pi_1 \succ \pi_2 \succ \ldots \succ \pi_{N_t})$ for the task-level scenario, and as $\pi^{N_t,k} = (\pi^k_1 \succ \pi^k_2 \succ \ldots \succ \pi^k_{N_t})$ for the instance-level scenario. Here, π_i represents the i-th best system according to the set S_{N_t} in the task-level scenario, while π^k_i represents the i-th best system according to π^k in the instance-level scenario.

Compatible Permutation When working with partial rankings, it is necessary to construct a complete ranking that respects the order of the evaluated systems, i.e., a linear extension of the partial ranking. This is accomplished by creating a compatible permutation (Gessel and Zhuang, 2018), which is a permutation of all systems consistent with the partial ranking. To construct a compatible permutation, we begin with the partial ranking of the evaluated systems and extend it to include the missing systems while maintaining the order of the evaluated systems. For example, let's consider a partial ordering $\pi_1 \succ \pi_2$ based on the evaluation of only these two systems. If there is an additional system that has not been evaluated, we can construct three compatible permutations: $\pi_3 \succ \pi_1 \succ \pi_2$, $\pi_1 \succ \pi_3 \succ \pi_2$ and $\pi_1 \succ \pi_2 \succ \pi_3$. These permutations ensure that the ordering of the evaluated systems is preserved while incorporating the missing system into the complete ranking.

Why use a combinatorial approach? Imputing missing data using compatible permutations enables us to leverage the Borda aggregation, inheriting its theoretical and practical advantages. Unlike classical methods like harmonic Fourier analysis (Kondor and Barbosa, 2010; Kondor and Dempsey, 2012; Clémençon et al., 2011) or multi-resolution analysis (Sibony et al., 2015), our approach works, providing a distinct combinatorial solution for imputing missing data in partial rankings.

3.2 Our Ranking Procedures

Our method can be described in two steps:

Our ranking procedure in a nutshell

1. **Matrix Representation of the rankings (Sssec. 3.2.1).** To harness the full potential of the available information in partial rankings, we **efficiently** generate all compatible permutations from the given partial rankings.

2. **Final System Ranking from Matrix Representation**. To obtain the final ranking of the systems, we propose a one-level (σ^l) approach (see Sssec. 3.2.2) for both task-level and instance-level information and a two-level aggregation approach (σ^{2l}) for instance-level information (see Sssec. 3.2.3).

3.2.1 Matrix representation of the rankings

Intuition. Our algorithm first summarizes the available information in all tasks and imputes the missing information in a consistent manner. To do this, we use a matrix representation M^π for each partial ranking π. This matrix decomposes the ranking information in pairwise variables, i.e., for every pair of systems i, j there is a variable representing the probability that system i outperforms system j.

Why using matrix representation? Using pairwise information has many advantages in ranking problems with missing data since it allows decomposing the total ranking information in $N(N-1)/2$ different variables. This decomposition has been used in statistical problems on partial and complete rankings (Fürnkranz and Hüllermeier, 2003; Lu and Boutilier, 2014a,b; Shah et al., 2017), for computing distances among partial rankings (Fagin et al., 2003), clustering (Ailon, 2010) and classification (Hüllermeier et al., 2008) among others. However, these problems consider specific forms of missing data such as top-k rankings (Fagin et al., 2003) or bucket orderings (Achab et al., 2019). Our approach differs from the aforementioned literature in the fact that we impute the missing data in a consistent manner in order to be able to deal with arbitrary missing data.

Efficiently building M^π. Let us consider a partial ranking π and let $M^\pi \in [0,1]^{N\times N}$ be its matrix representation. Matrix M^π_{ij} denotes the proportion of complete rankings that are compatible with π and satisfy the condition $i \succ j$, where i and j are distinct systems in the task. Formally, we can

distinguish three cases:

1. *if system i is directly compared to system j in π.* Set $M^{\pi}_{i,j} = 0 \; if \; i \succ j \; else \; M^{\pi}_{i,j} = 1$.

2. *if no information is provided for either system i or system j in π*, meaning that both systems are unobserved in the partial ranking. In this case, $M^{\pi}_{i,j} = 0.5$, which is the natural choice when no information is available.

3. *if we lack direct information about the comparison between system i and j in π (one system was evaluated and the was not)*, we represent this situation by setting the corresponding matrix entry to the proportion of compatible permutations ranking system i higher than system j among the total number of compatible permutations (see Ap. E).

A naive algorithm for generating the matrix M^{π} from π would have factorial complexity and it is thus exorbitant in practice for a relatively small number of systems, say $N > 10$. **One of the contributions of our solution is to reduce the complexity to $O(n^3)$ by efficiently computing** $M^{\pi}_{i,j}$. The closed-form expressions for $M^{\pi}_{i,j}$ as well as the proof for uniformity can be found in Ap. E.

3.2.2 σ^l: A one level approach

Intuition. At this stage, we have demonstrated the construction of a matrix M^{π} for a given partial ranking. However, in benchmarking scenarios, systems are typically evaluated on multiple tasks (in the case of task-level evaluation) or on multiple instances and tasks (in the case of instance-level evaluation), requiring the combination of multiple partial rankings. In this section, we will describe our approach for performing the one-level aggregation to address this requirement.

Combining Multiple Partial Rankings for Benchmarking. To combine the different matrices into a single matrix M we sum over all the tasks (in the case of task-level information) or instances and tasks (in the case of instance-level information). Formally, this is achieved by performing the following operation to obtain the combined matrix $M^I = \sum_{t\in[1,T]} \sum_{k\in[1,K_t]} M^{\pi^{r_t,k}}$, where $M^{\pi^{r_t,k}}$ represents the partial ranking induced on task t and instance k. Similarly, for the task level we define $M^T = \sum_{t\in[1,T]} M^{\pi_{r_t}}$ where $M^{\pi^{r_t}}$ represents the partial ranking induced on task t.

Obtaining the final system ranking In the final step, our goal is to obtain the final system ranking σ^l based on the matrix M^I or M^T. To achieve this, we use the Borda Count method, which involves computing the column-wise sum of the matrix and return the permutation that sorts the scores in increasing order. This step aligns with the approach proposed in (Colombo et al., 2022a). Formally:

$$\sigma^l = argsort\left(argsort\left[\textstyle\sum_i M_{i,0}, \cdots, \sum_i M_{i,N}\right]\right). \quad (3)$$

Here, M represents the matrix M^T for task-level information, and M^I for the instance-level.

3.2.3 σ^{2l}: A two-level approach

Intuition. In the case of instance-level information, we also present a two-step procedure that draws inspiration from the widely adopted two-step mean aggregation approach.

Procedure. In the first step, we apply the task-level aggregation approach to generate individual rankings for each task t, resulting in T different permutations. In the second step, we aggregate these multiple rankings using the Borda aggregation method. Formally σ^{2l} can be computed as:

1. For each task t, compute $M^t = \sum_{k\in[1,K_t]} M^{\pi^{r_t,k}}$

2. For each task t, compute $\sigma^{2l,t} = argsort\left(argsort\left[\sum_i M^t_{i,0}, \cdots, \sum_i M^t_{i,N}\right]\right)$.

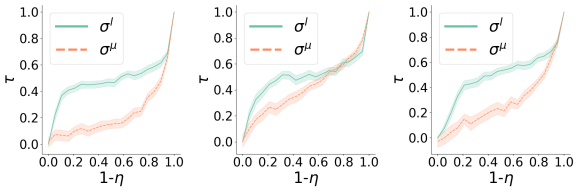
3. Compute the Borda count aggregation σ^{2l} of $[\sigma^{2l,1}, \cdots, \sigma^{2l,t}, \cdots, \sigma^{2l,T}]$.(Colombo et al., 2022b)

3.3 Confidence Intervals for σ^l

To evaluate systems with missing data, it is crucial to measure the uncertainty of partial rankings. In the previous section, we discussed combining partial rankings into a complete ranking. In this section, we analyze the confidence of our data regarding pairwise comparisons of system performance.

Under any ranking model such as Mallows Model (Fligner and Verducci, 1986) or Plackett-Luce (Plackett, 1975), M^{π}_{ij} are random variables of known expected value. In the previous section, we computed the empirical value $\widehat{M^{\pi}ij}$, approximating the true value $M^{\pi}ij$. Here, we want to know how close these two quantities are. Formally, we are looking for a confidence interval of level δ, that is the value for c_{ij} around $\widehat{M^{\pi}_{ij}}$ that contains M^{π}_{ij} with high probability, $P(|\widehat{M^{\pi}_{ij}} - M^{\pi}_{ij}| \geq c_{ij}) \leq 1 - \delta$. Noting that $0 \leq M^{\pi}_{ij} \leq 1$, we can use the Hoeffding inequality (Hoeffding, 1994) to compute the value of the confidence interval:

$$c_{ij} = \sqrt{\frac{-\log \delta}{2z_{ij}}}, \quad (4)$$



(a) GLUE (b) SGLUE (c) XTREM

Figure 2: Task-Level Robustness Experiment. We compare the robustness of our method σ^l with the mean aggregation method σ^μ by measuring the Kendall τ correlation coefficient between their respective rankings after removing a proportion η of scores and by considering the whole scores.

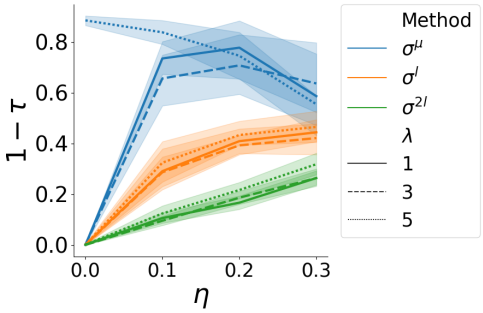


Figure 3: Synthetic experiment. Robustness for missing data η and different scaling corruptions λ.

where z_{ij} is the number of times the systems have been compared.

Intuition: to determine the significance of the difference in performance between system i and j, we can compare M_{ij} to 0.5. Thus, i performs better than j iff $M_{ij}^{\pi} > .5$. If the difference between M_{ij} and 0.5 is small, the performance difference between the two systems may not be statistically significant, indicating that we cannot determine which system performs better than the other.

The confidence interval developed above says that the true parameter M_{ij}^{π} is included in the interval $[\widehat{M_{ij}^{\pi}} - c_{ij}, \widehat{M_{ij}^{\pi}} + c_{ij}]$ with high probability. It follows that if 0.5 is not in this interval then we can say that one of the systems is better than the other with a high probability. Similar approaches have been proposed to find complete rankings and best-ranked systems with high probability (Busa-Fekete et al., 2014; Szörényi et al., 2015).

3.4 Baseline methods

To date, there is no established method for benchmarking NLP systems with missing data. To compare our proposed algorithm to existing methods, we consider a baseline approach that ignores missing data and relies on mean aggregation. This approach has been used in previous studies (Pfeiffer et al., 2022; Lin et al., 2022; Martin et al., 2020; Guibon et al., 2021; Peng et al., 2019), and we will refer to it as σ^μ in our experiments.

4 Synthetic Experiments

4.1 Data Generation

The analysis of a toy experiment involves synthetic scores generated from $N = 20$ systems, $T = 20$ tasks, and $K = 20$ instances. Each system's performance is modeled by a Gumbel random variable G_n with a center at $\phi \times n$ and a scale of $\beta = 1$, where ϕ is a dispersion parameter between 0 and 1. The scores of each system, $s(n, t, k)$, are independent and identically distributed samples of G_n centered at $\phi \times n$ with a scale of $\beta = 1$. Furthermore, the scores from different systems are sampled independently. Since the difference between G_{n+1} and G_n follows a logistic distribution with a mean of ϕ and a scale of 1, the probability that system $n + 1$ performs better than system n is at least 0.5, i.e., $P(G_{n+1} - G_n > 0) \geq 0.5$. Thus, the ranking of systems for all k and t is a realization of the true ranking $[1, \cdots, N]$, with a noise term controlled by the dispersion parameter ϕ. The extreme scenarios are $\phi = 0$ and $\phi = 1$, where $\phi = 0$ means that all scores $s(n, t, k)$ have the same distribution and $\phi = 1$ results in a strong consensus and a clear system ranking. Unless specifically mentioned, each experiment is repeated 100 times per data point.

4.2 Robustness To Scaling

In order to conduct a more detailed comparison of the ranking, we introduce a corruption in the scores of a specific task by rescaling them with a positive factor of λ. For this experiment, the corrupted tasks are randomly chosen. Although this corruption does not have any impact on our ranking process (since the ranking induced by a task-instance pair remains unchanged), it progressively disrupts the mean aggregation procedure as the value of λ increases (see Fig. 3 for detailed results). *This experiment further validates the use of rankings in NLP benchmarking, as these metrics involve different natures of measurements (e.g., BLEU score vs. number of parameters or speed) and can have bounded or unbounded scales.*

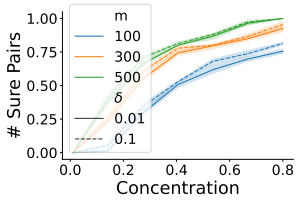
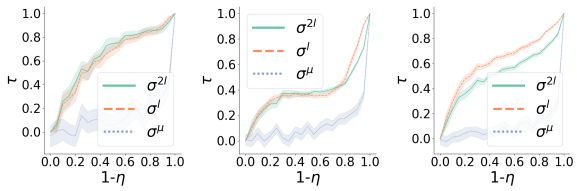


Figure 4: Confidence analysis.



(a) Dial. PC (b) COCO (c) SummEval

Figure 5: Instance-Level Robustness Experiment. We evaluate the robustness of our proposed aggregation methods, namely σ^{2l}, σ^{l}, and the mean aggregation method σ^{μ}, by randomly removing a proportion η of all instances on a specific task for a specific system.

4.3 Pairwise Confidence Analysis

To determine the number of system comparisons required to achieve a desired confidence level of δ, we use Eq. 4. Fig. 4 presents the results for two confidence levels (δ). The graph illustrates the number of system pairs for which 0.5 is not within the confidence interval, plotted against the number of comparisons for different values of m and ϕ. As expected, when the rankings are more concentrated (*i.e.*, when ϕ is closer to 1), fewer system comparisons are needed to achieve a high number of valid system comparisons. Real-world benchmarks usually have over 500 pairs.

5 Empirical Experiments

In this section, we benchmark our methods on real rankings. We introduce a dataset with over 100 million scores, surpassing previous datasets by several orders of magnitude (see Ssec. 5.1 and Ap. C).

5.1 A Comprehensive Collection of NLP System Scores

Our dataset builds upon the one used in (Colombo et al., 2022a) and includes two types of datasets: those with task-level information and those with instance-level information.

Datasets with Task Level Information Our datasets are based on GLUE (Wang et al., 2018), SGLUE (Wang et al., 2019), and XTREME (Hu et al., 2020), which include tasks of varying natures such as accuracy, F1-score, and mean square errors. In addition, we collected data from the GEM Benchmark (Gehrmann et al., 2021), which was an ideal use case for our methods as it encompasses missing data by design (as shown in Table 3 of (Gehrmann et al., 2021)) and includes evaluations of various natures such as lexical similarity, semantic equivalence, faithfulness evaluation, diversity, and system characterization (i.e., size of the vocabulary).

Datasets with Instance Level Information We did not use the data from (Peyrard et al., 2021) for the datasets with instance-level information because they did not provide the sentence and reference test required to add more evaluation metrics or more systems.

Therefore, we collected all the data from scratch and extended the dataset in two ways. First, we collected data from five distinct tasks - dialogue (Mehri and Eskenazi, 2020), image description (Young et al., 2014), summary evaluation (Dang et al., 2008; Owczarzak and Dang, 2011; Bhandari et al., 2020; Fabbri et al., 2021), data-to-text (Gardent et al., 2017; Zhou and Lampouras, 2020), and translation (Ranasinghe et al., 2021). For translation, we added datasets from WMT15 to WMT21 in several languages such as en, ru, ts, and others.

Secondly, we expanded the set of used metrics from 10 to 17, including Bleu (Papineni et al., 2002), MoverScore (Zhao et al., 2019) and BERTScore (Zhang et al., 2019a).

Overall, our benchmark grew *from 250K scores to over 131 M score. This extensive data work is one of the core contributions of this paper, and we believe it will be valuable for future research.* A more detailed description of the datasets can be found in Ap. C.

5.2 Task-Level Benchmarking in Real-World Scenarios

In this section, we explore aggregating missing data with task-level information. First, we test the robustness of our proposed method (σ^{l}) against the mean aggregation method (σ^{μ}) and then we quantify the difference between the two output rankings.

σ^{l} is more robust than σ^{μ}. To compare the effectiveness of aggregation methods in handling missing values, we randomly remove a proportion η of the task-level data and measure robustness by computing the Kendall τ between the rankings obtained

with and without missing values. From Fig. 2, we see that with no missing values (*i.e.*, $\eta = 0$), the aggregation methods yield the same rankings as the full data, with $\tau = 1$. Conversely, when all values are missing (*i.e.*, $\eta = 1$), there is no correlation. Overall, we find that σ^l achieves a higher correlation, with a large improvement of more than 10 points compared to other methods These results demonstrate that, on average, the rankings remain more stable when using our proposed method.

σ^l outputs a different ranking than σ^μ. We evaluated the correlation between rankings in the robustness experiment shown in Fig. 2. We compared the rankings from σ^l and σ^μ by computing the averaged τ across varying proportions of missing data. Results in Tab. 1 show a weak correlation between the rankings, indicating they produce different outcomes. This is further supported by Tab. 2, which shows the percentage of times the top 1 and top 3 rankings differ in the 2k generated rankings. These results demonstrate that our ranking procedure not only is more robust but also yields different conclusions when benchmarking systems with missing tasks.

	$\tau_{\sigma^l \leftrightarrow \sigma^\mu}$
GLUE	0.17 $\pm$0.24
SGLUE	0.33 $\pm$0.27
XTREM	0.26 $\pm$0.26
GEM	0.36 $\pm$0.36

Table 1: Agreement measured by Kendall τ correlation.

Dataset	top 1	top 3
GEM	0.52	0.25
SGLUE	0.20	0.15
GLUE	0.10	0.07
XTREM	0.19	0.09

Table 2: Percentage of times the top 1 and top 3 systems are the same between σ^l and σ^μ.

5.3 Instance-Level Benchmarking in Real-World Scenarios

In this section, we evaluate the robustness of σ^{2l}, σ^l, and the baseline σ^μ.

σ^{2l} and σ^l are more robust than σ^μ. Similarly to the previous robustness experiment, we randomly remove a proportion η of scores by discarding all instances of a specific task. *The goal of this missing value sampling is to simulate how missing scores may occur when certain systems are not evaluated on specific tasks*. For each method, Fig. 5 reports the τ correlation coefficient between the ranking obtained with missing values and the ranking obtained with complete scores.

Both σ^{2l} and σ^l produce highly correlated rankings, while being different from σ^μ. We conducted a replication of the agreement analysis presented in Ssec. 5.2 and present the findings in Tab. 3 and Tab. 4. Our results align with those of our previous experiments, demonstrating that both of our ranking-based procedures (σ^{2l} and σ^l) are more robust in the presence of missing data and yield different rankings than σ^μ.

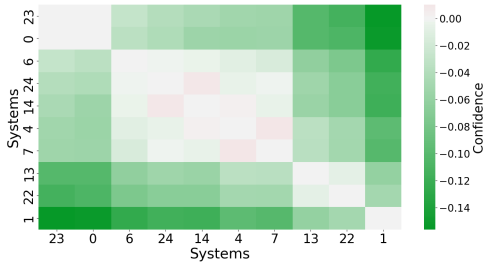


Figure 6: Confidence interval analysis on WMT en-de for a corruption level of $\eta = 0.2$ and a confidence level $\delta = 0.01$. The final ranking can be seen on the x-axis: left to right is best to worst

	Corr.
$\tau_{\sigma^{2l} \leftrightarrow \sigma^l}$	0.80 $\pm$0.22
$\tau_{\sigma^l \leftrightarrow \sigma^\mu}$	0.20 $\pm$0.28
$\tau_{\sigma^\mu \leftrightarrow \sigma^{2l}}$	0.19 $\pm$0.28

Table 3: Agreement.

	Top 1	Top 3
σ^{2l} vs σ^l	0.67	0.36
σ^l vs σ^μ	0.21	0.09
σ^μ vs σ^{2l}	0.19	0.09

Table 4: Top 1 & 3 analysis.

5.4 Statistical Analysis

Confidence interval for practitioners. The confidence interval is valuable for informing additional comparisons between systems i and j. A narrow interval indicates a reliable comparison, while a wider interval suggests more uncertainty and the need for additional comparisons across tasks. For example, in Fig. 6, we report the results of applying σ^l on WMT en-de with a confidence level of $\delta = 0.1$. Green value in position $i < j$ illustrate that system $0.5 \notin [\widehat{M}^\pi_{ij} - c_{ij}, \widehat{M}^\pi_{ij} + c_{ij}]$ and $i \succ j$ with high probability. The scale of green displays the distance between 0.5 and the CI, so the greener the more $i \succ j$. The results reveal distinct blocks where top systems (*i.e.,* 9,1,16,15) significantly outperform others with high confidence. Near the diagonal, the elements indicate relatively closer performance of the systems.

6 Conclusions

Our study sheds light on the limitations of the conventional mean-aggregation when dealing with missing data. Our novel statistical perspective and aggregation procedures that are both robust and grounded in social choice theory. Overral, the one-level aggregation method (σ^l) stands out as the most robust approach.

7 Limitations

The initial limitation we pinpoint is the task's reliance on the noise model applied to the data, which affects the outcomes. In an extreme scenario where a system lacks all measures, our method might not consistently rank it. Additional edge cases could be investigated, such as a system being poor in only one task with missing data, leading to potentially misleading ranking. To address this, we introduced the confidence interval in Section 3.3, supported by results in Section 5.4, to effectively recognize such challenging scenarios. It's important to highlight that these edge cases can impact all ranking procedures involving missing data.

Another limitation pertains to our ranking procedure's lack of consideration for user preferences regarding tasks. For instance, a user might emphasize certain tasks, such as A, D, and H, with task A carrying greater importance than the others. A natural approach to address this issue involves adopting a weighted variation of the Borda count or drawing inspiration from (Dwork et al., 2001). Although this avenue remains unexplored within our current work, it holds promise as a captivating direction for future investigations.

References

Mastane Achab, Anna Korba, and Stephan Clémençon. 2019. Dimensionality reduction and (bucket) ranking: a mass transportation approach. volume 98, pages 64–93. PMLR.

Nir Ailon. 2010. Aggregation of partial rankings, p-ratings and top-m lists. *Algorithmica (New York)*.

Khetam Al Sharou, Zhenhao Li, and Lucia Specia. 2021. Towards a better understanding of noise in natural language processing. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2021)*, pages 53–62.

Alnur Ali and Marina Meilă. 2012. Experiments with kemeny ranking: What works when? *Mathematical Social Sciences*, 64(1):28–40.

Vamsi Aribandi, Yi Tay, Tal Schuster, Jinfeng Rao, Huaixiu Steven Zheng, Sanket Vaibhav Mehta, Honglei Zhuang, Vinh Q Tran, Dara Bahri, Jianmo Ni, et al. 2021. Ext5: Towards extreme multi-task scaling for transfer learning. *arXiv preprint arXiv:2111.10952*.

Mikel Artetxe, Itziar Aldabe, Rodrigo Agerri, Olatz Perez-de Viñaspre, and Aitor Soroa. 2022. Does corpus quality really matter for low-resource languages? In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 7383–7390, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Mikel Artetxe, Sebastian Ruder, and Dani Yogatama. 2019. On the cross-lingual transferability of monolingual representations. *arXiv preprint arXiv:1910.11856*.

Mikel Artetxe and Holger Schwenk. 2019. Massively multilingual sentence embeddings for zero-shot cross-lingual transfer and beyond. *Transactions of the Association for Computational Linguistics*, 7:597–610.

Loïc Barrault, Ondřej Bojar, Marta R Costa-Jussa, Christian Federmann, Mark Fishel, Yvette Graham, Barry Haddow, Matthias Huck, Philipp Koehn, Shervin Malmasi, et al. 2019. Findings of the 2019 conference on machine translation (wmt19). ACL.

John J Bartholdi, Craig A Tovey, and Michael A Trick. 1989. The computational difficulty of manipulating an election. *Social Choice and Welfare*, 6(3):227–241.

D Bayer and P Diaconis. 1992. Trailing the dovetail shuffle to its lair. *The Annals of Applied Probability*, 2:294–313.

Luisa Bentivogli, Peter Clark, Ido Dagan, and Danilo Giampiccolo. 2009. The fifth pascal recognizing textual entailment challenge. In *TAC*.

Manik Bhandari, Pranav Gour, Atabak Ashfaq, Pengfei Liu, and Graham Neubig. 2020. Re-evaluating evaluation in text summarization. *arXiv preprint arXiv:2010.07100*.

Ondřej Bojar, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Shujian Huang, Matthias Huck, Philipp Koehn, Qun Liu, Varvara Logacheva, et al. 2017. Findings of the 2017 conference on machine translation (wmt17). Association for Computational Linguistics.

Ondřej Bojar, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Philipp Koehn, Varvara Logacheva, Christof Monz, et al. 2016. Findings of the 2016 conference on machine translation. In *Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers*, pages 131–198.

Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. 2021. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Róbert Busa-Fekete, Eyke Hüllermeier, and Balázs Szörényi. 2014. Preference-Based Rank Elicitation using Statistical Models: The Case of Mallows. In *Proceedings of the 31th International Conference on Machine Learning, (ICML)*, pages 1071–1079.

Daniel Cer, Mona Diab, Eneko Agirre, Inigo Lopez-Gazpio, and Lucia Specia. 2017. Semeval-2017 task 1: Semantic textual similarity-multilingual and cross-lingual focused evaluation. *arXiv preprint arXiv:1708.00055*.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. Boolq: Exploring the surprising difficulty of natural yes/no questions. *arXiv preprint arXiv:1905.10044*.

Jonathan H Clark, Eunsol Choi, Michael Collins, Dan Garrette, Tom Kwiatkowski, Vitaly Nikolaev, and Jennimaria Palomaki. 2020. Tydi qa: A benchmark for information-seeking question answering in typologically diverse languages. *Transactions of the Association for Computational Linguistics*, 8:454–470.

Stéphan Clémençon, Romaric Gaudel, and Jérémie Jakubowicz. 2011. Clustering rankings in the fourier domain. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2011, Athens, Greece, September 5-9, 2011. Proceedings, Part I 11*, pages 343–358. Springer.

Pierre Colombo, Nathan Noiry, Ekhine Irurozki, and Stephan CLEMENCON. 2022a. What are the best systems? new perspectives on NLP benchmarking. In *Advances in Neural Information Processing Systems*.

Pierre Colombo, Nathan Noiry, Ekhine Irurozki, and Stéphan Clémençon. 2022b. What are the best systems? new perspectives on NLP benchmarking. In *NeurIPS*.

Pierre Colombo, Maxime Peyrard, Nathan Noiry, Robert West, and Pablo Piantanida. 2022c. The glass ceiling of automatic evaluation in natural language generation. *arXiv preprint arXiv:2208.14585*.

Alexis Conneau, Guillaume Lample, Ruty Rinott, Adina Williams, Samuel R Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. Xnli: Evaluating cross-lingual sentence representations. *arXiv preprint arXiv:1809.05053*.

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2005. The pascal recognising textual entailment challenge. In *Machine Learning Challenges Workshop*, pages 177–190. Springer.

Hoa Trang Dang, Karolina Owczarzak, et al. 2008. Overview of the tac 2008 update summarization task. In *TAC*.

Marie-Catherine De Marneffe, Mandy Simons, and Judith Tonhauser. 2019. The commitmentbank: Investigating projection in naturally occurring discourse. In *proceedings of Sinn und Bedeutung*, volume 23, pages 107–124.

Mostafa Dehghani, Yi Tay, Alexey A Gritsenko, Zhe Zhao, Neil Houlsby, Fernando Diaz, Donald Metzler, and Oriol Vinyals. 2021. The benchmark lottery. *arXiv preprint arXiv:2107.07002*.

William B Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*.

Cynthia Dwork, Ravi Kumar, Moni Naor, and Dandapani Sivakumar. 2001. Rank aggregation methods for the web. In *Proceedings of the 10th international conference on World Wide Web*, pages 613–622.

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.

Ronald Fagin, Ravi Kumar, and D. Sivakumar. 2003. Comparing top k lists. *SIAM Journal on Discrete Mathematics*.

Angela Fan, Shruti Bhosale, Holger Schwenk, Zhiyi Ma, Ahmed El-Kishky, Siddharth Goyal, Mandeep Baines, Onur Celebi, Guillaume Wenzek, Vishrav Chaudhary, et al. 2021. Beyond english-centric multilingual machine translation. *The Journal of Machine Learning Research*, 22(1):4839–4886.

Akhbardeh Farhad, Arkhangorodsky Arkady, Biesialska Magdalena, Bojar Ondřej, Chatterjee Rajen, Chaudhary Vishrav, Marta R Costa-jussa, España-Bonet Cristina, Fan Angela, Federmann Christian, et al. 2021. Findings of the 2021 conference on machine translation (wmt21). In *Proceedings of the Sixth Conference on Machine Translation*, pages 1–88. Association for Computational Linguistics.

Michael A Fligner and Joseph S Verducci. 1986. Distance based ranking models. *Journal of the Royal Statistical Society*, 48(3):359–369.

Johannes Fürnkranz and Eyke Hüllermeier. 2003. Pairwise preference learning and ranking. pages 145–156.

Claire Gardent, Anastasia Shimorina, Shashi Narayan, and Laura Perez-Beltrachini. 2017. The webnlg challenge: Generating text from rdf data. In *Proceedings of the 10th International Conference on Natural Language Generation*, pages 124–133.

Sebastian Gehrmann, Tosin Adewumi, Karmanya Aggarwal, Pawan Sasanka Ammanamanchi, Aremu Anuoluwapo, Antoine Bosselut, Khyathi Raghavi Chandu, Miruna Clinciu, Dipanjan Das, Kaustubh D Dhole, et al. 2021. The gem benchmark: Natural language generation, its evaluation and metrics. *arXiv preprint arXiv:2102.01672*.

Sebastian Gehrmann, Abhik Bhattacharjee, Abinaya Mahendiran, Alex Wang, Alexandros Papangelis, Aman Madaan, Angelina McMillan-Major, Anna Shvets, Ashish Upadhyay, Bingsheng Yao, et al. 2022a. Gemv2: Multilingual nlg benchmarking in a single line of code. *arXiv preprint arXiv:2206.11249*.

Sebastian Gehrmann, Elizabeth Clark, and Thibault Sellam. 2022b. Repairing the cracked foundation: A survey of obstacles in evaluation practices for generated text. *arXiv preprint arXiv:2202.06935*.

Ira M Gessel and Yan Zhuang. 2018. Shuffle-compatible permutation statistics. *Advances in Mathematics*, 332:85–141.

Danilo Giampiccolo, Bernardo Magnini, Ido Dagan, and William B Dolan. 2007. The third pascal recognizing textual entailment challenge. In *Proceedings of the ACL-PASCAL workshop on textual entailment and paraphrasing*, pages 1–9.

Gaël Guibon, Matthieu Labeau, Hélène Flamein, Luce Lefeuvre, and Chloé Clavel. 2021. Few-shot emotion recognition in conversation with sequential prototypical networks. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6858–6870, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Ralf Herbrich, Tom Minka, and Thore Graepel. 2006. Trueskill™: A bayesian skill rating system. In *Advances in Neural Information Processing Systems*, volume 19. MIT Press.

Wassily Hoeffding. 1994. Probability inequalities for sums of bounded random variables. *The collected works of Wassily Hoeffding*, pages 409–426.

Junjie Hu, Sebastian Ruder, Aditya Siddhant, Graham Neubig, Orhan Firat, and Melvin Johnson. 2020. Xtreme: A massively multilingual multi-task benchmark for evaluating cross-lingual generalisation. In *International Conference on Machine Learning*, pages 4411–4421. PMLR.

Dieuwke Hupkes, Mario Giulianelli, Verna Dankers, Mikel Artetxe, Yanai Elazar, Tiago Pimentel, Christos Christodoulopoulos, Karim Lasri, Naomi Saphra, Arabella Sinclair, et al. 2022. State-of-the-art generalisation research in nlp: a taxonomy and review. *arXiv preprint arXiv:2210.03050*.

Eyke Hüllermeier, Johannes Fürnkranz, Weiwei Cheng, and Klaus Brinker. 2008. Label ranking by learning pairwise preferences. *Artificial Intelligence*, 172:1897–1916.

Daniel Khashabi, Snigdha Chaturvedi, Michael Roth, Shyam Upadhyay, and Dan Roth. 2018. Looking beyond the surface: A challenge set for reading comprehension over multiple sentences. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 252–262.

Donald E Knuth. 1970. Permutations, matrices and generalized young tableaux. *Pacific Journal of Mathematics*, 34:709–727.

Jan Kocoń, Igor Cichecki, Oliwier Kaszyca, Mateusz Kochanek, Dominika Szydło, Joanna Baran, Julita Bielaniewicz, Marcin Gruza, Arkadiusz Janz, Kamil Kanclerz, Anna Kocoń, Bartłomiej Koptyra, Wiktoria Mieleszczenko-Kowszewicz, Piotr Miłkowski, Marcin Oleksy, Maciej Piasecki, Łukasz Radliński, Konrad Wojtasik, Stanisław Woźniak, and Przemysław Kazienko. 2023. Chatgpt: Jack of all trades, master of none. *Preprint*, arXiv:2302.10724.

R Kondor and M Barbosa. 2010. Ranking with kernels in fourier space.

Risi Kondor and Walter Dempsey. 2012. Multiresolution analysis on the symmetric group. *Advances in Neural Information Processing Systems*, 25.

Eric Lehman, Evan Hernandez, Diwakar Mahajan, Jonas Wulff, Micah J. Smith, Zachary Ziegler, Daniel Nadler, Peter Szolovits, Alistair Johnson, and Emily Alsentzer. 2023. Do we still need clinical language models? *Preprint*, arXiv:2302.08091.

Hector Levesque, Ernest Davis, and Leora Morgenstern. 2012. The winograd schema challenge. In *Thirteenth International Conference on the Principles of Knowledge Representation and Reasoning*.

Patrick Lewis, Barlas Oğuz, Ruty Rinott, Sebastian Riedel, and Holger Schwenk. 2019. Mlqa: Evaluating cross-lingual extractive question answering. *arXiv preprint arXiv:1910.07475*.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Chin-Yew Lin, Guihong Cao, Jianfeng Gao, and Jian-Yun Nie. 2006. An information-theoretic approach to automatic evaluation of summaries. In *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*, pages 463–470.

Stephanie Lin, Jacob Hilton, and Owain Evans. 2021. Truthfulqa: Measuring how models mimic human falsehoods. *arXiv preprint arXiv:2109.07958*.

Xi Victoria Lin, Todor Mihaylov, Mikel Artetxe, Tianlu Wang, Shuohui Chen, Daniel Simig, Myle Ott, Naman Goyal, Shruti Bhosale, Jingfei Du, Ramakanth Pasunuru, Sam Shleifer, Punit Singh Koura, Vishrav Chaudhary, Brian O'Horo, Jeff Wang, Luke Zettlemoyer, Zornitsa Kozareva, Mona Diab, Veselin Stoyanov, and Xian Li. 2022. Few-shot learning with multilingual generative language models. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 9019–9052, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Barrault Loïc, Biesialska Magdalena, Bojar Ondřej, Federmann Christian, Graham Yvette, Grundkiewicz Roman, Haddow Barry, Huck Matthias, Joanis Eric, Kocmi Tom, et al. 2020. Findings of the 2020 conference on machine translation (wmt20). In *Proceedings of the Fifth Conference on Machine Translation*, pages 1–55. Association for Computational Linguistics,.

Tyler Lu and Craig Boutilier. 2014a. Effective sampling and learning for mallows models with pairwise-preference data. *Journal of Machine Learning Research*.

Tyler Lu and Craig Boutilier. 2014b. Effective sampling and learning for mallows models with pairwise-preference data. *Journal of Machine Learning Research*.

Louis Martin, Benjamin Muller, Pedro Javier Ortiz Suárez, Yoann Dupont, Laurent Romary, Éric de la Clergerie, Djamé Seddah, and Benoît Sagot. 2020. CamemBERT: a tasty French language model. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7203–7219, Online. Association for Computational Linguistics.

Shikib Mehri and Maxine Eskenazi. 2020. Usr: An unsupervised and reference free evaluation metric for dialog generation. *arXiv preprint arXiv:2005.00456*.

Jun-Ping Ng and Viktoria Abrecht. 2015. Better summarization evaluation with word embeddings for rouge. *arXiv preprint arXiv:1508.06034*.

Joakim Nivre, Mitchell Abrams, Zeljko Agic, Lars Ahrenberg, Lene Antonsen, et al. 2018. Universal dependencies 2.2 (2018). *URL http://hdl. handle. net/11234/1-1983xxx. LINDAT/CLARIN digital library at the Institute of Formal and Applied Linguistics, Charles University, Prague, http://hdl. handle. net/11234/1-1983xxx*.

Jekaterina Novikova, Ondřej Dušek, and Verena Rieser. 2018. Rankme: Reliable human ratings for natural language generation. *arXiv preprint arXiv:1803.05928*.

OpenAI. 2023. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

Karolina Owczarzak and Hoa Trang Dang. 2011. Overview of the tac 2011 summarization track: Guided task and aesop task. In *Proceedings of the Text Analysis Conference (TAC 2011), Gaithersburg, Maryland, USA, November*.

Xiaoman Pan, Boliang Zhang, Jonathan May, Joel Nothman, Kevin Knight, and Heng Ji. 2017. Cross-lingual name tagging and linking for 282 languages. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1946–1958.

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.

Yifan Peng, Shankai Yan, and Zhiyong Lu. 2019. Transfer learning in biomedical natural language processing: An evaluation of BERT and ELMo on ten benchmarking datasets. In *Proceedings of the 18th BioNLP Workshop and Shared Task*, pages 58–65, Florence, Italy. Association for Computational Linguistics.

Maxime Peyrard, Teresa Botschen, and Iryna Gurevych. 2017. Learning to score system summaries for better content selection evaluation. In *Proceedings of the Workshop on New Frontiers in Summarization*, pages 74–84.

Maxime Peyrard, Wei Zhao, Steffen Eger, and Robert West. 2021. Better than average: Paired evaluation of nlp systems. *arXiv preprint arXiv:2110.10746*.

Jonas Pfeiffer, Naman Goyal, Xi Lin, Xian Li, James Cross, Sebastian Riedel, and Mikel Artetxe. 2022. Lifting the curse of multilinguality by pre-training modular transformers. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3479–3495, Seattle, United States. Association for Computational Linguistics.

Mohammad Taher Pilehvar and Jose Camacho-Collados. 2018. Wic: the word-in-context dataset for evaluating context-sensitive meaning representations. *arXiv preprint arXiv:1808.09121*.

Robin L Plackett. 1975. The analysis of permutations. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 24(2):193–202.

Maja Popović. 2017. chrf++: words helping character n-grams. In *Proceedings of the second conference on machine translation*, pages 612–618.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Belgium, Brussels. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551.

Afshin Rahimi, Yuan Li, and Trevor Cohn. 2019. Massively multilingual transfer for ner. *arXiv preprint arXiv:1902.00193*.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*.

Tharindu Ranasinghe, Constantin Orasan, and Ruslan Mitkov. 2021. An exploratory analysis of multilingual word-level quality estimation with cross-lingual transformers. *arXiv preprint arXiv:2106.00143*.

Machel Reid and Mikel Artetxe. 2022. PARADISE: Exploiting parallel data for multilingual sequence-to-sequence pretraining. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 800–810, Seattle, United States. Association for Computational Linguistics.

Ond rej Bojar, Christian Federmann, Mark Fishel, Yvette Graham, Barry Haddow, Matthias Huck, Philipp Koehn, and Christof Monz. 2018. Findings of the 2018 conference on machine translation (wmt18). In *Proceedings of the Third Conference on Machine Translation*, volume 2, pages 272–307.

Mario Rodríguez-Cantelar, Chen Zhang, Chengguang Tang, Ke Shi, Sarik Ghazarian, João Sedoc, Luis Fernando D'Haro, and Alexander Rudnicky. 2023. Overview of robust and multilingual automatic evaluation metrics for open-domain dialogue systems at dstc 11 track 4. *arXiv preprint arXiv:2306.12794*.

Melissa Roemmele, Cosmin Adrian Bejan, and Andrew S Gordon. 2011. Choice of plausible alternatives: An evaluation of commonsense causal reasoning. In *2011 AAAI Spring Symposium Series*.

Sebastian Ruder. 2021. Challenges and Opportunities in NLP Benchmarking. `http://ruder.io/nlp-benchmarking`.

João Sedoc and Lyle Ungar. 2020. Item response theory for efficient human evaluation of chatbots. In *Proceedings of the First Workshop on Evaluation and Comparison of NLP Systems*, pages 21–33, Online. Association for Computational Linguistics.

Nihar B. Shah, Sivaraman Balakrishnan, Adityanand Guntuboyina, and Martin J. Wainwright. 2017. Stochastically transitive models for pairwise comparisons: Statistical and computational issues. *IEEE Transactions on Information Theory*.

Eric Sibony, Stéphan Clémençon, and Jérémie Jakubowicz. 2015. Mra-based statistical learning from incomplete rankings. In *International Conference on Machine Learning*, pages 1432–1441. PMLR.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642.

Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, Abu Awal Md Shoeb, Abubakar Abid, Adam Fisch, Adam R Brown, Adam Santoro, Aditya Gupta, Adrià Garriga-Alonso, et al. 2022. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *arXiv preprint arXiv:2206.04615*.

Lars St, Svante Wold, et al. 1989. Analysis of variance (anova). *Chemometrics and intelligent laboratory systems*, 6(4):259–272.

R P Stanley. 1986. *Enumerative Combinatorics*. Wadsworth Publishing Company.

Miloš Stanojević, Amir Kamran, Philipp Koehn, and Ondřej Bojar. 2015. Results of the wmt15 metrics shared task. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 256–273.

Balázs Szörényi, Róbert Busa-Fekete, Adil Paul, and Eyke Hüllermeier. 2015. Online rank elicitation for plackett-luce: A dueling bandits approach. *Advances in Neural Information Processing Systems*, 28.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2019. Superglue: A stickier benchmark for general-purpose language understanding systems. *arXiv preprint arXiv:1905.00537*.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*.

Alex Warstadt, Amanpreet Singh, and Samuel R Bowman. 2019. Neural network acceptability judgments. *Transactions of the Association for Computational Linguistics*, 7:625–641.

Herbert S. Wilf. 1999. *East Side, West Side . . . - an introduction to combinatorial families-with Maple programming*. arxiv.

Adina Williams, Nikita Nangia, and Samuel R Bowman. 2017a. A broad-coverage challenge corpus for sentence understanding through inference. *arXiv preprint arXiv:1704.05426*.

Adina Williams, Nikita Nangia, and Samuel R Bowman. 2017b. A broad-coverage challenge corpus for sentence understanding through inference. *arXiv preprint arXiv:1704.05426*.

Yinfei Yang, Yuan Zhang, Chris Tar, and Jason Baldridge. 2019. Paws-x: A cross-lingual adversarial dataset for paraphrase identification. *arXiv preprint arXiv:1908.11828*.

Peter Young, Alice Lai, Micah Hodosh, and Julia Hockenmaier. 2014. From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions. *Transactions of the Association for Computational Linguistics*, 2:67–78.

Sheng Zhang, Xiaodong Liu, Jingjing Liu, Jianfeng Gao, Kevin Duh, and Benjamin Van Durme. 2018. Record: Bridging the gap between human and machine commonsense reading comprehension. *arXiv preprint arXiv:1810.12885*.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019a. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*.

Yuan Zhang, Jason Baldridge, and Luheng He. 2019b. Paws: Paraphrase adversaries from word scrambling. *arXiv preprint arXiv:1904.01130*.

Wei Zhao, Maxime Peyrard, Fei Liu, Yang Gao, Christian M Meyer, and Steffen Eger. 2019. Moverscore: Text generation evaluating with contextualized embeddings and earth mover distance. *arXiv preprint arXiv:1909.02622*.

Giulio Zhou and Gerasimos Lampouras. 2020. Webnlg challenge 2020: Language agnostic delexicalisation for multilingual rdf-to-text generation. In *Proceedings of the 3rd International Workshop on Natural Language Generation from the Semantic Web (WebNLG+)*, pages 186–191.

Pierre Zweigenbaum, Serge Sharoff, and Reinhard Rapp. 2017. Overview of the second bucc shared task: Spotting parallel sentences in comparable corpora. In *Proceedings of the 10th Workshop on Building and Using Comparable Corpora*, pages 60–67.

Pierre Zweigenbaum, Serge Sharoff, and Reinhard Rapp. 2018. Overview of the third bucc shared task: Spotting parallel sentences in comparable corpora. In *Proceedings of 11th Workshop on Building and Using Comparable Corpora*, pages 39–42.

Appendices

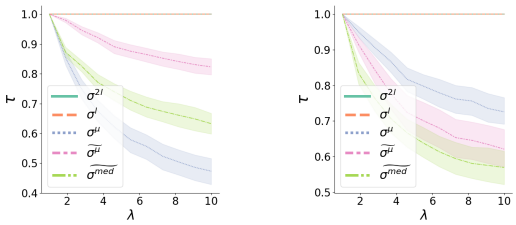
A Extended Related Work and Other Baselines methods

A.1 More on missing data in NLP.

Another approach to handling missing data in benchmarks would be to create new datasets, however, it can be a sluggish, costly, and expertise-demanding process (see footnote 5 in (Lin et al., 2021)). Moreover, there are situations where collection becomes infeasible, such as when working with private datasets, calling for the need to develop tools that can rank systems with missing scores.

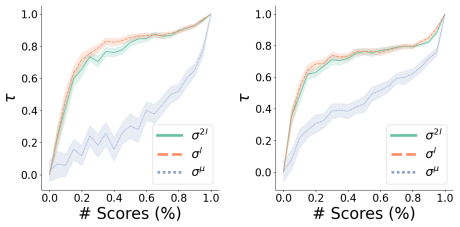
A.2 Why not directly imputing data in the score?

Directly imputing values as scores is not the current practice in NLP. In fact, this approach would be inadequate due to potential variations in metric scale and difficulty, leading to a failure in accurately capturing task difficulty (as mentioned above and in (Dolan and Brockett, 2005)). To illustrate this to we present some experiments.



(a) WMT 2018 en-ru (b) WMT 2019 zh-en

Figure 7: **Imputation methods are not robust to scaling**. To further compare the ranking, we corrupt the scores of a given task by re-scaling them by a factor λ. Whereas it does not affect our ranking procedure (every ranking induced by a task-instance pair remains the same), it increasingly perturbs the mean aggregation and other imputation procedures as λ increases. $\tilde{\sigma^{\text{med}}}$ corresponds to the median imputation and $\tilde{\sigma^{\mu}}$ corresponds to the mean imputation.

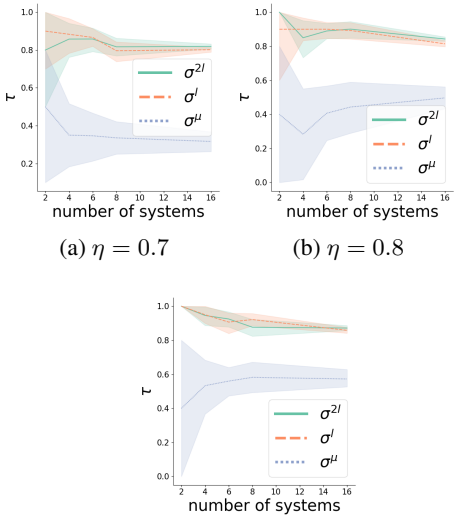


(a) WMT 2018 en-ru (b) WMT 2019 zh-en

Figure 8: **Additional Instance-Level Robustness Experiment (see Figure 17)**. We evaluate the robustness of our proposed aggregation methods, namely σ^{2l}, σ^{l}, and the mean aggregation method σ^{μ}, by randomly removing a proportion η of all instances on a specific task for a specific system.

A.3 Ablation experiments

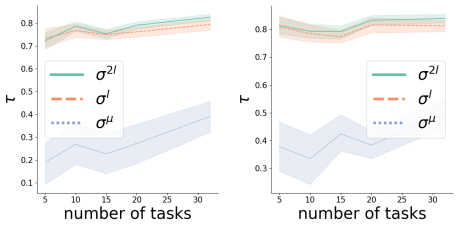
An interesting experiment is checking how the performance of the different methods vary when changing the number if systems, tasks or utterances. In the following experiments, we take the WebNLG2020 English dataset. For each ablation experiment—whether it's concerning systems, tasks, or metrics—and for each η value within the set $0.7, 0.8, 0.9$, we randomly select the systems, tasks, or utterances to retain. Then, in the same way as the robustness experiment, we compute correlations and iterate this process 20 times. Results are presented in Figures 9, 12 and 11. As we can see, the correlations are consistent when changing the number of these parameters.



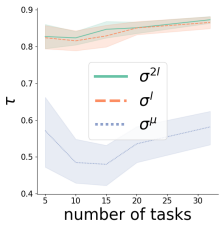
(a) $\eta = 0.7$ (b) $\eta = 0.8$

(c) $\eta = 0.9$

Figure 9: Ablation experiment on the number of systems

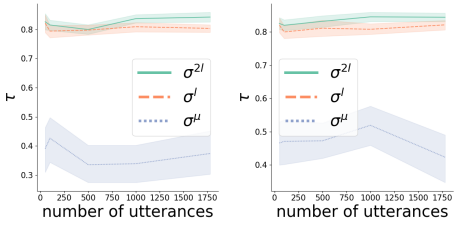


(a) $\eta = 0.7$ (b) $\eta = 0.8$

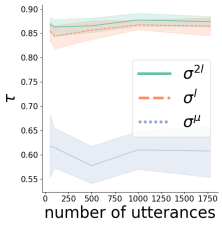


(c) $\eta = 0.9$

Figure 10: Ablation experiment on the number of tasks



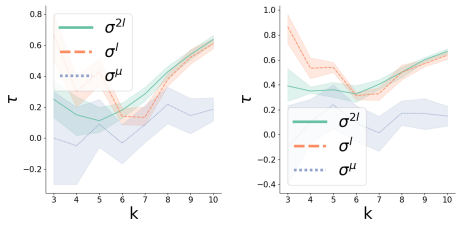
(a) $\eta = 0.7$ (b) $\eta = 0.8$



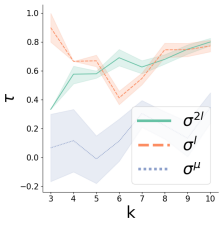
(c) $\eta = 0.9$

Figure 11: Ablation experiment on the number of utterances

Another similar experiment we investigate here is when only keeping the top k systems (determined for each aggregation method using the complete dataset). Here are the results :



(a) $\eta = 0.7$ (b) $\eta = 0.8$



(c) $\eta = 0.9$

Figure 12: Ablation experiment on the top k systems

A.4 Other ideas of future work

In the future, we would like to explore several refinements of the method:

- **Impact of the inter-task correlation**. The task correlation can impact the choice of the best system. In the future, we would like to study the impact of the choice of the ranking procedure in depth.
- **Impact of misleading evaluation**. Evaluation in NLP can be noisy due to the variety in language and lack of metric robustness (Al Sharou et al., 2021; Rodríguez-Cantelar et al., 2023). Future work will include the consideration of this factor when choosing the aggregation method.
- **Comparison with the ANOVA method (St et al., 1989)**. Although this is slightly outside the scope of the paper, we would like to compare our confidence interval with the one obtained with the ANOVA method.

A.5 More on the technical contribution of the algorithm.

Our technical contribution boils down to extending the Borda aggregation to the case of missing data (aka incomplete rankings). In the ranking literature, two types of approaches can be identified to deal with partial rankings: Relying on top-k rankings. In this case, all the systems are evaluated but only those that are ranked in the first k positions are provided. There are many methods to aggregate

and all in this setting, for example, (Ailon, 2010). This is different from our scenario where some systems cannot be evaluated on particular tasks. Relying on incomplete rankings. In this case, only k systems are evaluated on a specific task. This fits our scenario. Rank aggregation/statistical analysis in the case of k=2 is called pairwise ranking and is well handled by the literature (Knuth, 1970; Lu and Boutilier, 2014a; Plackett, 1975; Popović, 2017; Zhang et al., 2018). These approaches are limited and only use pairwise comparisons which can lead to paradoxes when ranking more systems. In this paper, we introduce an aggregation procedure for arbitrary values of k. Our main technical contribution is to extend the Borda aggregation to incomplete rankings. To the best of our knowledge, this is the only paper dealing with aggregation -not specifically Borda- of incomplete rankings.

B Ethical Statement & Limitation of our work

It is important to consider the potential ethical implications and limitations of our work. One ethical concern is the potential bias in the reranking process, as the selection of the "best" hypothesis may favor certain perspectives or reinforce existing biases present in the training data. Care should be taken to ensure fairness and mitigate any potential bias before applying our methods.

C Dataset Description

C.1 Task Level Information

We provide additional details on the data collection for Task Level Information.

We gathered data from four benchmark studies, namely GLUE (General Language Understanding Evaluation) (Wang et al., 2018), SGLUE (SuperGLUE) (Wang et al., 2019)[1], XTREME (Hu et al., 2020) and GEM. In the GLUE dataset, there were a total of 105 systems evaluated across nine different tasks: CoLA, SST-2, MRPC, STS-B, QQP, MNLI, QNLI, RTE, and WNLI (Warstadt et al., 2019; Socher et al., 2013; Dolan and Brockett, 2005; Cer et al., 2017; Rajpurkar et al., 2016; Williams et al., 2017a; Dagan et al., 2005; Giampiccolo et al., 2007; Bentivogli et al., 2009; Levesque et al., 2012). The SGLUE dataset consisted of 24 systems evaluated on 10 different tasks: BoolQ, CB, COPA, MultiRC, ReCoRD, RTE, WiC, WSC, AX-b, and AX-g (Clark et al., 2019; De Marneffe et al., 2019; Roemmele et al., 2011; Khashabi et al., 2018; Zhang et al., 2018; Levesque et al., 2012; Pilehvar and Camacho-Collados, 2018). The XTREME benchmark comprised 15 systems and included tasks such as sentence classification (XNLI and PAXS-X), structured prediction (Universal Dependencies v2.5 and Wikiann), sentence retrieval (BUCC and Tatoeba), and question answering (XQuAD, MLQA, TyDiQA-GoldP) (Conneau et al., 2018; Williams et al., 2017b; Yang et al., 2019; Zhang et al., 2019b; Nivre et al., 2018; Rahimi et al., 2019; Pan et al., 2017; Zweigenbaum et al., 2018, 2017; Artetxe and Schwenk, 2019; Artetxe et al., 2019; Rajpurkar et al., 2016; Lewis et al., 2019; Clark et al., 2020).

Each benchmark employed a variety of metrics with different scales, including accuracy, f1, and correlation. Additionally, the GEM benchmark involved 22 systems evaluated using diverse metrics such as prediction length, vocabulary size, entropy, Rouge, NIST, Bleu', Meteor', Bleurt, Nubia, and Bertscore.

C.2 Instance Level Information

In this particular setting, our primary focus is on evaluating the performance of natural language generation (NLG) systems, as these scores are among the easiest to collect. We concentrate on five different tasks: summary evaluation, image description, dialogue, and translation. For *summary evaluation*, we utilize the TAC08 (Dang et al., 2008), TAC10, TAC11 (Owczarzak and Dang, 2011), RSUM (Bhandari et al., 2020), and SEVAL (Fabbri et al., 2021) datasets. Regarding *sentence-based image description*, we rely on the FLICKR dataset (Young et al., 2014). For *dialogue*, we make use of the PersonaChat (PC) and TopicalChat (TC) datasets (Mehri and Eskenazi, 2020). For the translation part, we added datasets from WMT15 (Stanojević et al., 2015), WMT16 (Bojar et al., 2016), WMT17 (Bojar et al., 2017), WMT18 (rej Bojar et al., 2018), WMT19 (Barrault et al., 2019), WMT20 (Loïc et al., 2020), and WMT21 (Farhad et al., 2021) in several languages such as en, ru, ts, and others. For all datasets except MLQE, we consider automatic metrics based on S3 (both variant pyr/resp) (Peyrard et al., 2017), ROUGE (Lin, 2004) (including five of its variants (Ng and Abrecht, 2015)), JS [1-2] (Lin et al., 2006), Chrfpp (Popović, 2017), BLEU, BERTScore (Zhang et al., 2019a), and MoverScore (Zhao et al., 2019). For

[1]Results can be accessed at https://super.gluebenchmark.com/

the MLQE dataset, we solely consider several versions of BERTScore, MoverScore, and ContrastScore. Additionally, we incorporate human evaluation, which is specific to each dataset.

C.3 Data Statistics

To give to the reader a better sense of the richness of our benchmark, we report in Fig. 13 the statistics on our dataset. We demonstrate a diverse distribution of system counts across various datasets, ranging from a minimum of 2 systems to a maximum of 60 systems. Regarding the total number of sentences (instances) and the average number per system, as depicted in Fig. 14 and Fig. 15, the smaller datasets consist of several hundred sentences in total, while the larger datasets encompass up to several hundred thousand sentences in total.

D Additional Real-Data Experiments

In this dedicated section, we aim to provide curious readers with a deeper understanding of the capabilities of our methods by presenting additional figures and experimental results. Through these supplementary materials, we intend to shed more light on the effectiveness and potential of our approaches, enabling readers to gain valuable insights into our methods.

D.1 Example of Ranking with missing data on XTREM

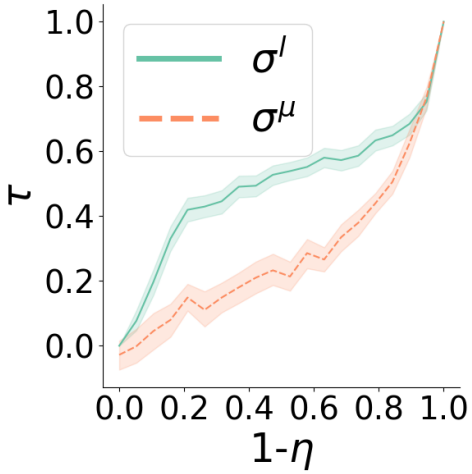
In this section, we aim to illustrate the distinction between different rankings obtained using σ^l and σ^μ on XTREM dataset for a specific noise realization. Using Tab. 5, we obtain the following rankings:

- σ^l gives the following ranking : $M0 > M3 > M2 > M1 > M7 > M5 > M4 > M8 > M9$
- σ^μ gives the following ranking : $M7 > M4 > M0 > M6 > M9 > M2 = M3 > M1 > M8 > M5$.

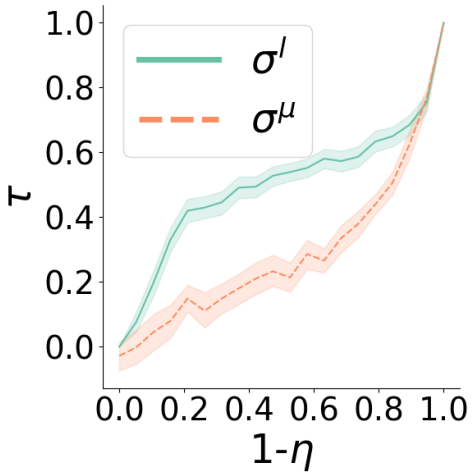
We can see that the two methods disagree on the best systems in this case. However, as can be seen in our experiments, the ranking-based method is more robust.

D.2 Additional Robustness Experiment on task level datasets

In this section, we report additional experiments on the task level robustness.



(a) XTREM



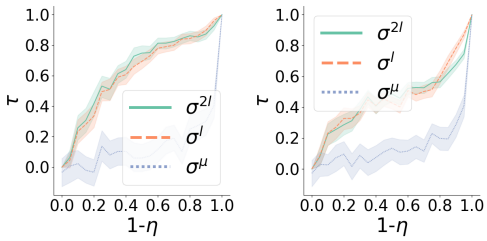
(b) GEM

Figure 16: Robustness experiments on XTREM and GEM

Model	Classification	Structured Prediction	Question Answering	Sentence Retrieval
M0	90.3	X	76.3	93.7
M1	90.1	X	75.0	X
M2	89.3	75.5	75.2	92.4
M3	89.0	76.7	73.4	93.3
M4	88.3	X	X	X
M5	X	X	X	X
M6	87.9	75.6	X	91.9
M7	X	X	X	92.6
M8	X	75.4	X	X
M9	88.2	74.6	X	89.0

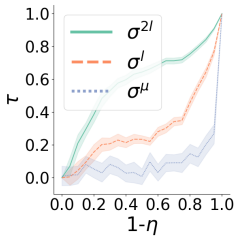
Table 5: XTREM dataset with 10 systems and 18 missing values ($\eta = 0.45$)

D.3 Additional Robustness Experiment on instance level datasets

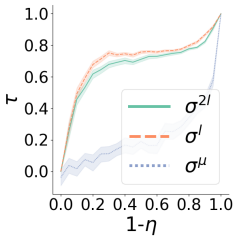


(a) Dialogue PC

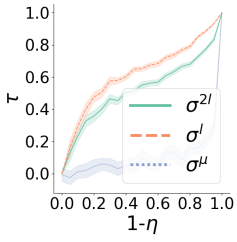
(b) Dialogue TC



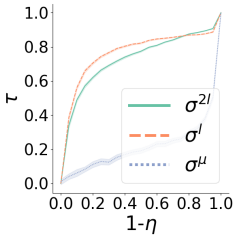
(c) Flickr



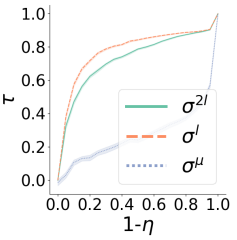
(d) COCO



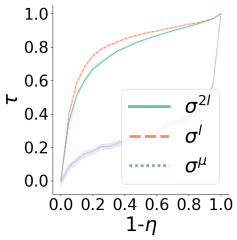
(e) SummEval



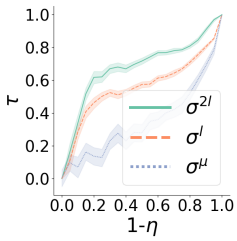
(f) TAC 08



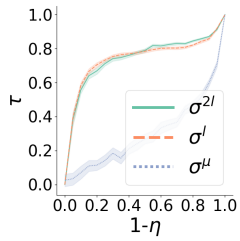
(g) TAC 09



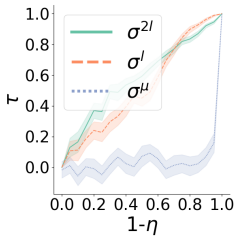
(h) TAC 11



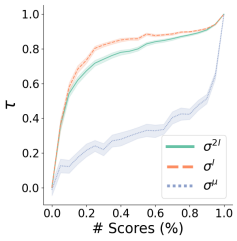
(i) WebNLG2017



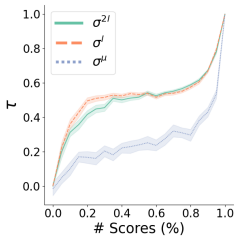
(j) WebNLG2020 en



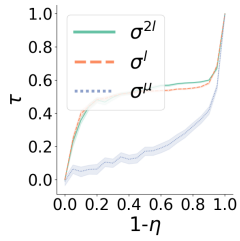
(k) WebNLG2020 ru



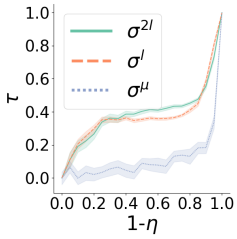
(l) WMT20 cs-en



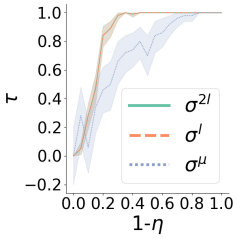
(m) WMT20 pl-en



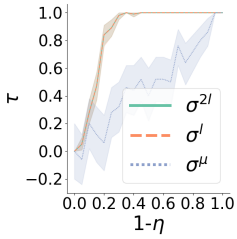
(n) WMT21 en-de



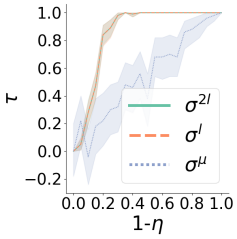
(o) WMT21 en-ru



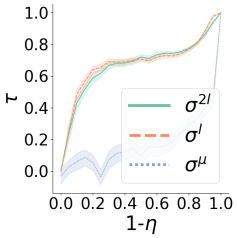
(p) WMT21 challengeset de-en



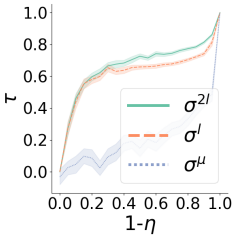
(q) WMT21 challengeset en-de



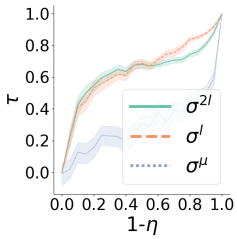
(r) WMT21 challengeset zh-en



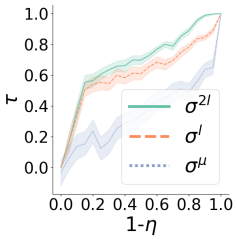
(s) WMT21 florestest bn-hi



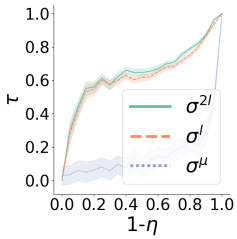
(t) WMT21 florestest hi-bn



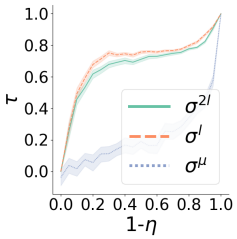
(u) WMT21 florestest xh-zu



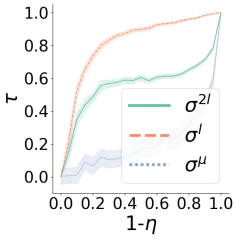
(v) WMT21 florestest zu-xh



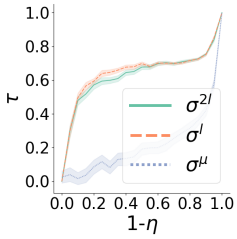
(w) WMT21 cs-en



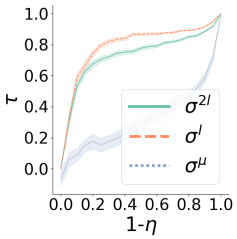
(x) WMT21 en-cs



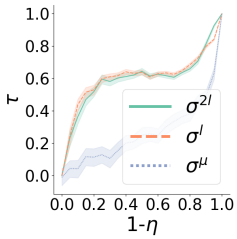
(y) WMT21 de-fr



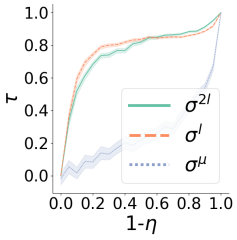
(z) WMT21 en-ha



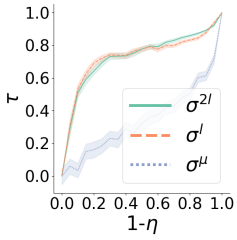
(aa) WMT21 en-is



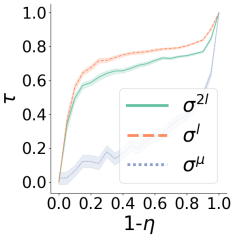
(ab) WMT21 fr-de



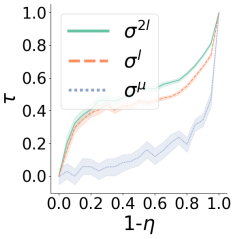
(ac) WMT21 ha-en



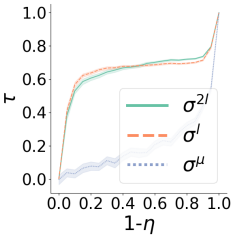
(ad) WMT21 is-en



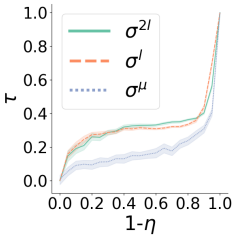
(ae) WMT21 ja-en



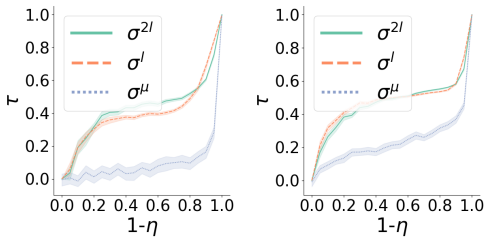
(af) WMT21 ru-en



(ag) WMT21 zh-en



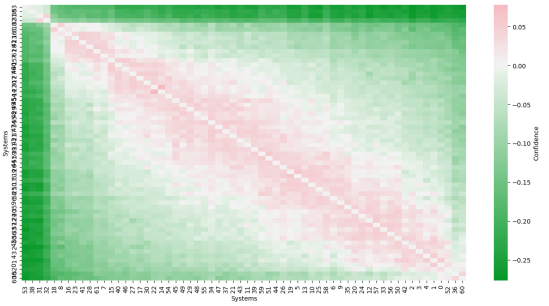
(ah) WMT21 tedtalks en-de



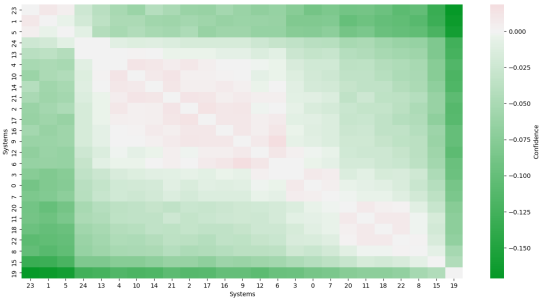
(ai) WMT21 tedtalks en-ru

(aj) WMT21 tedtalks zh-en

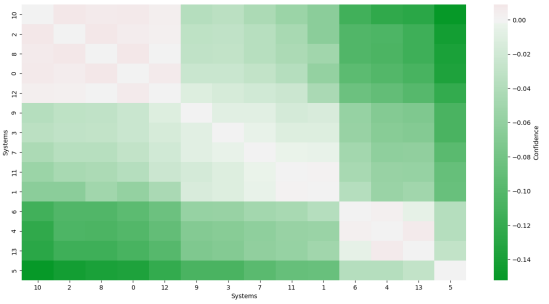
Figure 17: Instance-Level Robustness Experiment. We evaluate the robustness of our proposed aggregation methods, namely σ^{2l}, σ^{l}, and the mean aggregation method σ^{μ}, by randomly removing a proportion η of all instances on a specific task for a specific system. Each experiment is repeated 100 times for each proportion.



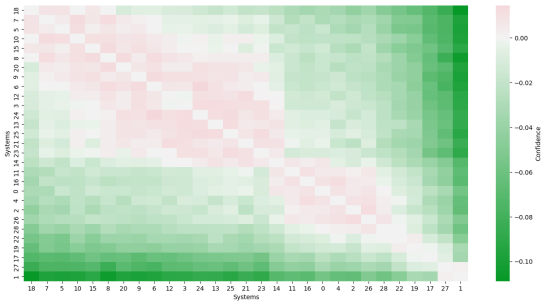
(ak) TAC08



(al) WMT21 en-de



(am) WMT21 en-ha



(an) WMT21 en-zh

Figure 18: Confidence intervals for various instance level datasets with $\eta = 0.2$ and $\delta = 0.01$

D.4 Additional Confidence Analysis on Task Level

In this section, we present additional experiments conducted on four instance-level datasets. We computed confidence intervals for the instance-level, similar to the approach used in Section **??**. Consistent with the main findings in the paper, our observations reveal that closer performance among systems is indicated near the diagonal and we can clearly observe group of systems. This analysis of confidence intervals provides valuable insights into the relative performance of different systems.

E On the Rankings

This section gathers technical considerations on the ranking methods used in our algorithm.

E.1 Borda Count on permutations (in vector notation)

Remark 2. *The Borda count is a ranking system that aggregates a set of permutations* $\sigma^1, \ldots, \sigma^L \in$

$\mathfrak{S}_N$ *by summing the ranks of each system and then ranking the obtained sums. The procedure is as follows:*

1. Compute $\text{sum}_n := \sum_{l=1}^{L} \sigma_n^l$ *for every* $1 \leq n \leq N$,

2. Output $\sigma := \text{Borda}(\sigma^1, \ldots, \sigma^L) \in \mathfrak{S}_N$ *that ranks the sums,* sum_n *(*$\text{argsort}(\text{argsort}(\text{sum}_1, \ldots, \text{sum}_T))$*).*

E.2 Borda Count on permutations in pairwise matrix notation

In Sssec. 3.2.1 we argue that a ranking $\sigma \in \mathfrak{S}_N$ can also be written as a pairwise matrix and in Sssec. 3.2.2 and Sssec. 3.2.3 we further elaborate on how to write ranking data-set D in pairwise matrix form $M^D \in [0,1]^{N \times N}$. Under this notation, the final aggregated ranking σ for the Borda count algorithm can be shown to be equivalent to the permutation that sorts the sum of the columns in M^D,

$$\sigma = argsort\left(argsort\left[\textstyle\sum_i M^D_{i,0}, \cdots, \sum_i M^D_{i,N}\right]\right). \quad (5)$$

E.3 Generating all compatible rankings

In this section, we detail the computation of the $M^\pi_{i,j}$ when item i is not evaluated and item j is evaluated. Let us fix some notation first. For the following, k is the number of observed systems in π, item i is not evaluated, item j is evaluated and r is the (partial) rank of item j. Under this setting, we set $M^\pi_{i,j} = p(n,k,r)$, i.e., the proportion of compatible rankings that rank i before j when π has k items. The closed-form expressions for these quantities are given in Eq. 6. Here we note that $t(n,k)$ is the total number of rankings of n items compatible with π, S^a_b is the number of shuffles of two lists of lengths a and b and V^a_b denotes the variations of a out of b items, i.e., the number of possible arrangements of selections of a objects out of b, where the order of the selected objects matters.

$$\begin{aligned} p(n,k,r) = &\sum_{i=0}^{n-k-1} V^i_{n-k-1} * (i+1) \\ &* S^r_{i+1}(n-k-i-1)! \\ &* S^{n-k-i-1}_{k-r-1}/t(n,k) \\ t(n,k) = &(n-k)! * S^k_{n-k} \\ S^a_b = &(a+b)!/(a!+b!) \\ V^a_b = &a!/(b-a)! \end{aligned} \quad (6)$$

Remark 3. *A naive algorithm for generating the matrix* M^π *from* $\sigma \in S_{N-r_{tk}}$ *would have factorial complexity and it is thus exorbitant in practice for a relatively small number of systems, say* $N > 10$*. However, our solution has a complexity of* $O(n^3)$ *and can be precomputed once at the beginning of the benchmarking process to efficiently generate the pairwise matrix* M^π *from partial ranking* π*.*

E.4 Proof of uniformity

In this section, we give the intuition and the proof for Eq. 6. This section follows a classic strategy on Enumerative Combinatorics (Stanley, 1986; Wilf, 1999): if we can define an algorithm to generate compatible permutations uniformly at random (such as that in Algorithm 2), we can easily adapt it to count those permutations to yield an efficient counting expression, as we do in Eq. 6.

We start by introducing 2 basic operations of $permute$ and $shuffle$, along with the number of possible outcomes of these operations.

Permute a list - $permute(l)$ Given a list of n objects, generate a permutation of these items. There are $n!$ possible ways of permuting n items. An efficient way for generating random permutations is the Fisher-Yates-Knuth algorithm (Knuth, 1970).

Shuffle two lists - $shuffle(A,B)$ Given two disjoint lists of distinct elements A, B of lengths a, b respectively, generate a permutation σ of the two lists of length $a+b$ in such a way that the relative order of the items in the lists A and B is respected in σ. This name and idea is based on the popular way of shuffling two decks of cards (Bayer and Diaconis, 1992). Its easy to see that Algorithm **??** generates every possible shuffling with equal probability. The total number of shuffles of lists A, B is given in Eq. 6 as S^a_b.

Counting complete, compatible rankings At this point, we are ready to detail the expression of $p(n,k,r)$ in Eq. 6, both the intuition and the proof of uniformity. For this, we propose in Algorithm 2

Algorithm 1: Generate a random shuffle of lists A and B

1 **for** $i \in [a+b]$ **do**
2 $rand \leftarrow$ random number in $[0,1]$;
3 **if** $rand > 0.5 \vee B$ *is empty* $\wedge A$ *is non empty* **then**
4 $\sigma(i) = pop(A)$;
5 **else**
6 $\sigma(i) = pop(B)$;
7 **end**
8 **end**

Algorithm 2: Generate a random ranking among those compatible with β

1 **for** $s \in [n]$ **do**
2 $\eta_{head} \leftarrow s-1$ items from η where the order matters ;
3 $\eta_{head} \leftarrow$ insert i in η_{head} ;
4 $\sigma_{head} \leftarrow$ shuffle$(\eta_{head}, \beta_{head})$;
5 $\eta_{tail} \leftarrow \eta \setminus \eta_{head}$;
6 $\eta_{tail} \leftarrow$ permute(η_{tail}) ;
7 $\sigma_{tail} \leftarrow$ shuffle$(\eta_{tail}, \beta_{tail})$;
8 **return** $(\sigma_{head} \succ j \succ \sigma_{tail})$;
9 **end**

to sample complete, compatible rankings and then adapt this sampling algorithm to a counting algorithm in Theorem 1.

Notation We start by fixing the notation. Let β be a partial ranking of length k which includes item j in rank r, $\beta_1 \succ \ldots \succ \beta_r = j \succ \ldots \succ \beta_k$. Let η be a disjoint set of $n-k$ items that have not been ranked and which includes the unobserved item i. The goal is to generate (i) a compatible ranking with β (a ranking σ of all the items in such a way that the relative ordering of the items of β is maintained) and (ii) which ranks item i before item j. We denote the "s-head" of a list to the items in the first s positions in that list.

Intuition We are now ready to explain the intuition. Each of the possible compatible permutations that rank i before j is generated in the following way:

Algorithm 2 generates permutations that rank item j at position s, item i before j and we iterate for all possible values of s. First, in line line 2 we select $s-1$ items randomly from η, where the order of the items matter (i.e., a variation). Then, we insert item i in a random position of this list, denoted η_{head} in line 3. In line 4 we shuffle these two lists, i.e., η_{head} and the $r-$head of β, β_{head}, i.e., the sublist with the items that are ranked before j. The result of the shuffling process is the $s+r$-head of the output permutation σ. We permute the rest of the unobserved items denoting these list η_{tail}, in line 6. Finally, we shuffle this list η_{tail} and the $k-r$-tail of η in line 7. The result of this shuffle is the tail of σ. Finally, in line 8 we return the concatenation of $\sigma_{head}, j, \sigma_{tail}$, which is clearly a compatible permutation with β as the relative order of the items in β is maintained in the output.

It is easy to see that Algorithm 2 generates the target permutations uniformly at random. Following a classic strategy on Enumerative Combinatorics (Stanley, 1986; Wilf, 1999) we use this algorithm as a proof for $p(n,k,r)$.

Theorem 1. *The number of complete permutations of n items compatible with partial ranking β that rank the unobserved item i before the observed item j is given by the following expression,*

$$p(n,k,r) = \sum_{i=0}^{n-k-1} V_{n-k-1}^{i} * (i+1) \quad (7)$$

$$* S_{i+1}^{r}(n-k-i-1)! \quad (8)$$

$$* S_{k-r-1}^{n-k-i-1}/t(n,k). \quad (9)$$

Proof. It is easy to see that in Algorithm 2 there is a bijection between the permutations in the target (that is, the permutations compatible with β for which $i \succ j$) and each outcome of Algorithm 2. Clearly, for uniform at random outcomes of the $shuffle$ and $permute$ operations, the outcome of Algorithm 2 will be random as well. Therefore, the number of possible outcomes of the algorithm equals the number of permutations in the target.

It follows that each term in $p(n,k,r)$ Each term in the previous expression comes from a different line in 2:

- Line 2: The number of variations of i items out of $n-k-1$ is V_{n-k-1}^{i}.
- Line 3: There are $s+1$ ways of inserting item i, thus the term $(r+1)$.
- Line 4: There are S_{s+1}^{r} ways of shuffling η_{head} and β_{head}.

- Line 6: There are $(n-k-s-1)!$ possible permutations of the items in η_{tail}.
- Line 7: There are $S_{k-r-1}^{n-k-s-1}$ ways of shuffling the two tails.
- Line 8: Finally, since we compute the proportion by dividing among the total number of compatible permutations.

By repeating this process for all $s < n-k-1$ the proof is completed.

□

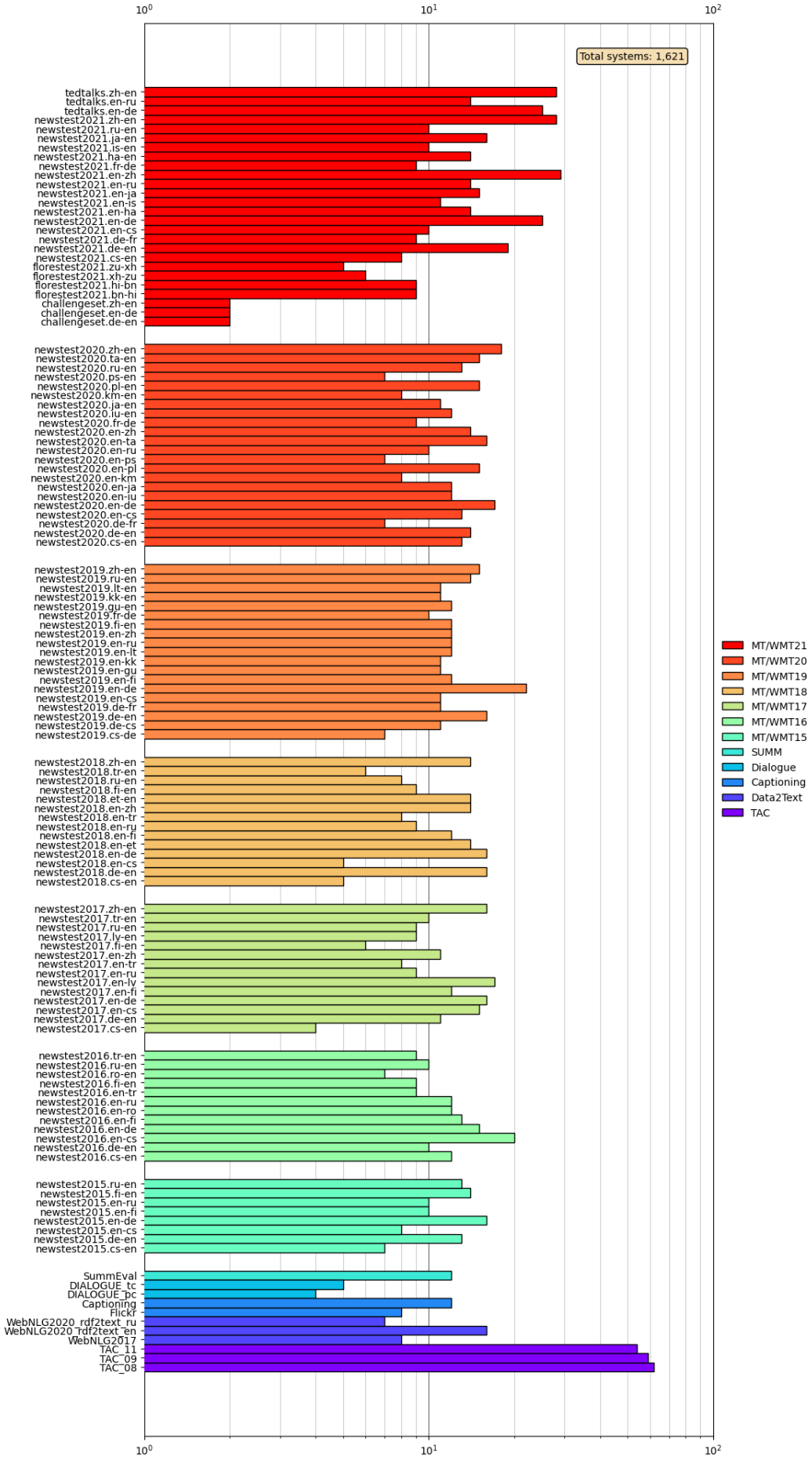


Figure 13: Number of systems in each dataset (log scale)

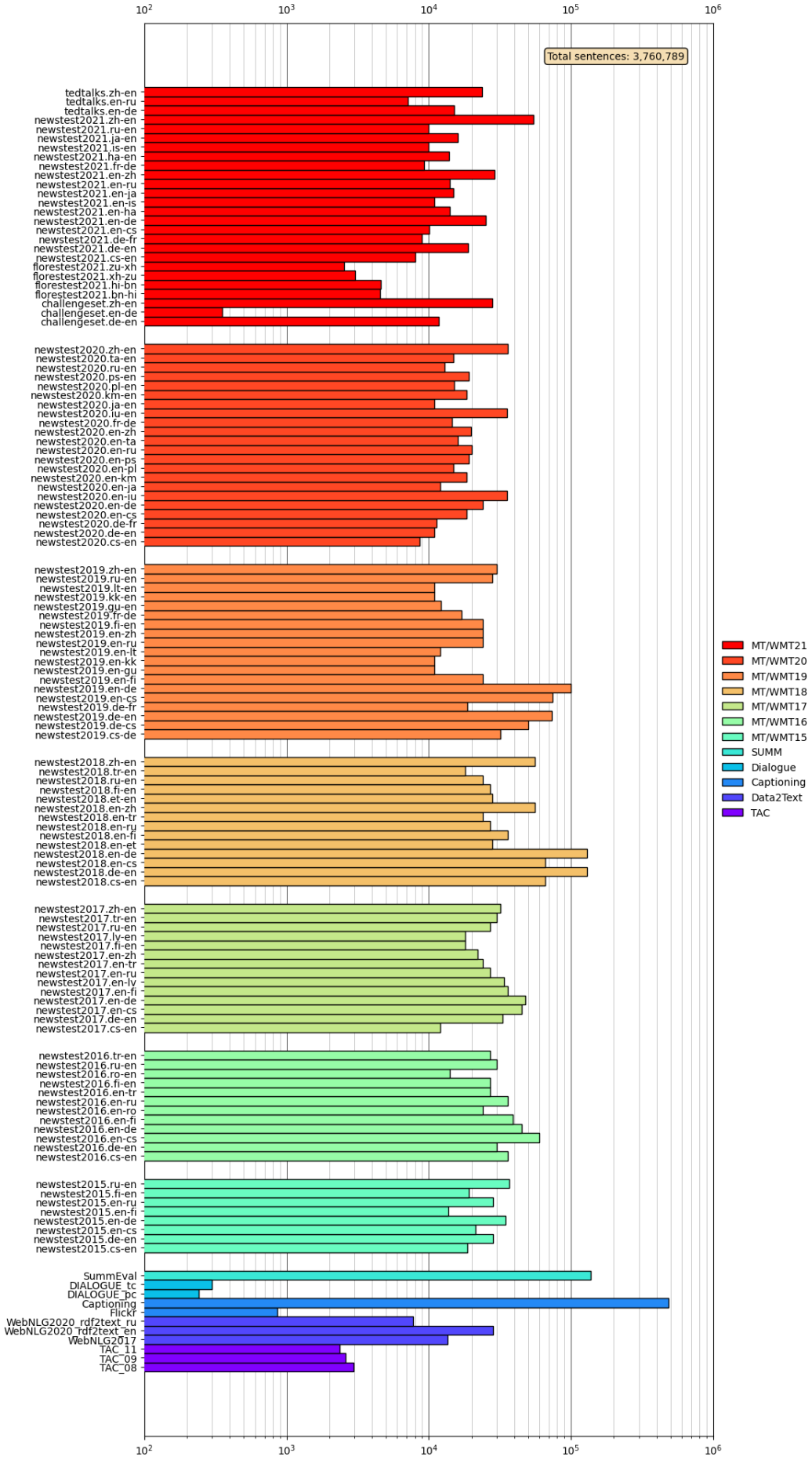


Figure 14: Number of sentences in each dataset (log scale)

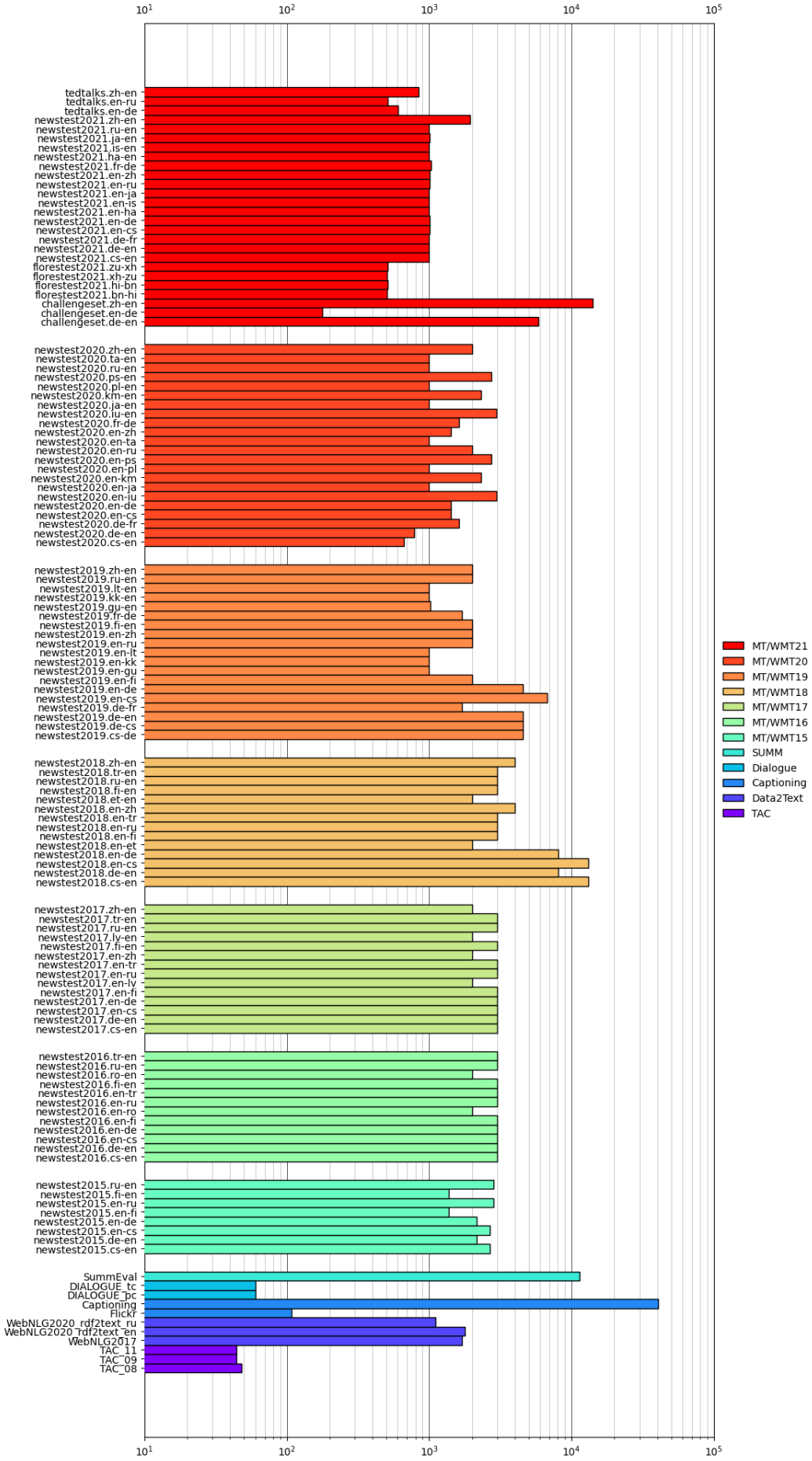


Figure 15: Average number of sentences per system in each dataset (log scale)