



**HAL**  
open science

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

Pierre Colombo, Anas Himmi, Ekhine Irurozki, Nathan Noiry, Stéphan Cléménçon

► **To cite this version:**

Pierre Colombo, Anas Himmi, Ekhine Irurozki, Nathan Noiry, Stéphan Cléménçon. Towards More Robust NLP System Evaluation: Handling Missing Scores in Benchmarks. 2024. hal-04575102

**HAL Id: hal-04575102**

**<https://hal.science/hal-04575102v1>**

Preprint submitted on 14 May 2024

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

---

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

---

**Anas Himmi\***  
MICS, CentraleSupélec  
anas.himmi@student-cs.fr

**Ekhine Irurozki\***  
S2A, Telecom Paris  
ekhine.irurozki@telecom-paris.fr

**Nathan Noiry**  
Owkin  
noiry.nathan@gmail.com

**Stephan Clemençon**  
S2A, Telecom Paris  
stephan.clemençon@telecom-paris.fr

**Pierre Colombo\***  
MICS, CentraleSupélec  
pierre.colombo@centralesupelec.fr

## 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 [112, 44, 102]. However, evaluating general-purpose systems, such as foundation models used for generative tasks [74?, 91, 17, 103], presents unique challenges. A single task, metric, or dataset may not be sufficient to effectively gauge their capabilities [62, 90, 113]. Therefore, it is crucial to develop tools that can benchmark these systems on a multitude of tasks [5], enabling a comprehensive assessment of their overall performance [97].

In recent years, the field of natural language processing (NLP) has made significant strides, with frequent emergence of new models [74, 71, 17, 91, 103, 80, 50] and techniques [16, 65]. To evaluate the performance of these systems across various tasks, datasets, and metrics [33] have

---

\*main authors

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 [119]), high costs associated with reproducing experiments (*e.g.*, see Table 3 in [6]), incomplete datasets (see Table 5 in [109]), data collection errors, data cleaning procedures, data privacy concerns (particularly in-house datasets [61]), and specialized expertise required to process niche datasets [95]. In recent work, two main approaches have been followed to deal with missing scores, which are discarding data [98] or ignoring certain tasks (see Table 10 in [79] and Table 5 in [84]) or evaluations. However, these approaches are unsatisfactory as they can lead to biased and unreliable evaluations.

In this work, we aim to address the challenge of 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 [98, 79, 84, 61, 95].
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 method includes a novel combinatorial approach for inputting 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 comprehensive and accurate evaluation of NLP systems:** previous works on score aggregation relied on a benchmark of 250K scores [31, 97], and did not release 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, yielding different results. These highlight the importance of considering missing system scores in evaluation.

In order to promote the adoption of our method, we will publicly share the code and data at <https://github.com/AnasHimmi/MissingDataRanking>, enabling researchers to apply our approach and assess NLP system performance more reliably.

## 1.1 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. Mean aggregation has commonly been adopted to consolidate information at both the instance and task levels.

**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 task 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 [57, 56, 58]. An illustration of the general framework (*i.e.*, with instance and system level) for benchmarking can be found in Fig. 1.

## 1.2 Problem Formulation

The notation used in this discussion will closely follow that of a previously mentioned [31]. In essence, we are dealing with a scenario where  $N$  systems are being compared based on their performance

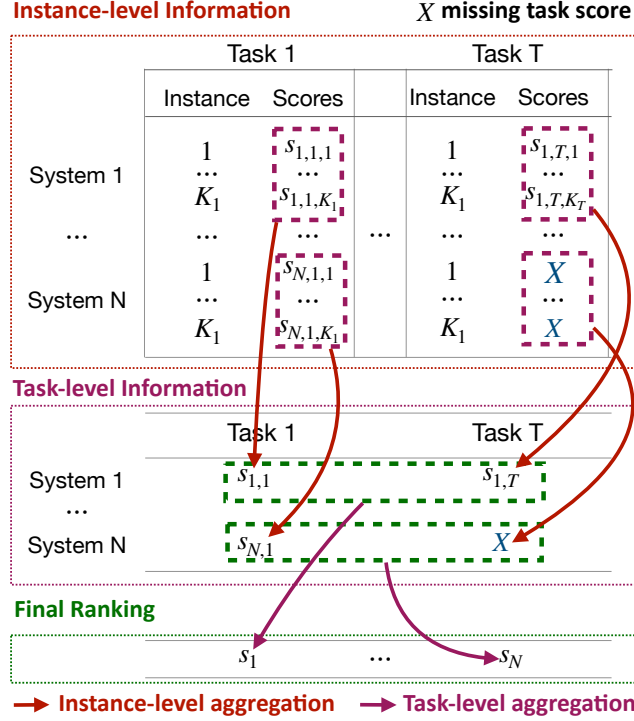


Figure 1: **Framework for benchmarking NLP systems with two information granularity:** *instance-level* (red above) and *task-level* (purple below). The final goal of benchmarking is to produce a ranking (green bottom). The instance-level aggregation allows for the derivation of task-level information,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

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 a  $k$  of 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 systems 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  $\sigma_i$ -th best system according to the considered aggregation. Equivalently, ordering  $\pi = (\pi_1 \succ \pi_2 \succ \dots \succ \pi_N)$  denotes that  $\pi_i$  is better than system  $\pi_{i+1}$  for all  $i$ . Let us first defined the different granularity of benchmarking depending whether we have access to individual 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  $f^T$ :

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

where  $\mathcal{S}_{N_t} = (s_{n,t}, 1 \leq n \leq N_t)$  is the set of score achieved by each system evaluated on the task  $t$ . Note that only  $N_t$  systems are evaluate 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  $f^I$ :

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

where  $\mathcal{S}_{N_i}^k = (s_{n,t,k}, 1 \leq n \leq N_i)$  is the set of 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 [31], we have  $N_t = N$  for all  $t \in [1, T]$ .

### 1.3 Handling Complete Scores in NLP System Evaluation

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

**Mean aggregation** ( $\sigma^\mu$ ) is the default choice for practitioners. At the task level  $\sigma^\mu$  is defined as  $\sigma^\mu = \text{argsort} \left( \text{argsort} \left[ \frac{1}{T} \sum_{1 \leq t \leq T} s_{n,t} \text{ for } 1 \leq n \leq N \right] \right)$  and at the instance level  $\sigma^\mu = \text{argsort} \left( \text{argsort} \left[ \frac{1}{T} \sum_{1 \leq t \leq T} \frac{1}{K_t} \sum_{1 \leq k \leq K_t} s_{n,t,k} \text{ for } 1 \leq n \leq N \right] \right)$ , where  $\text{argsort}(\mathbf{u})$  is the permutation that sorts the items in  $\mathbf{u}$ . 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, metric 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 [97, 31]. These methods aggregate rankings instead of scores. In [31], the authors tackle the problem of generating a ranking by aggregating rankings, utilizing the Borda count method (see Ssec. D.1 for more details on Borda Count) known for its computational properties [10, 47, 3]. Extending the Borda count method is not a straightforward task either. In the next section, we will present our aggregation procedure that can handle missing system score on a whole task.

## 2 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.

### 2.1 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  $S_{N_t}^k$ . 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  $\pi^{N_t} = (\pi_1 \succ \pi_2 \succ \dots \succ \pi_{N_t})$  for the task-level scenario, and as  $\pi^{N_t,k} = (\pi_1^k \succ \pi_2^k \succ \dots \succ \pi_{N_t}^k)$  for the instance-level scenario. Here,  $\pi_i$  represents the  $i$ -th best system according to the set  $S_{N_t}$  in the task-level scenario, while  $\pi_i^k$  represents the  $i$ -th best system according to  $\pi^k$  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 [59], 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 using a combinatorial approach?** Inputting missing data using compatible permutations enables us to leverage the widely used Borda method for aggregation, inheriting its theoretical and practical advantages. Unlike classical methods like harmonic Fourier analysis [72, 73, 25] or multi-resolution analysis [116], our approach works, providing a distinct combinatorial solution for inputting missing data in partial rankings.

## 2.2 Our Ranking Procedures: from scores to system ranking

This section describes our algorithms for benchmarking when there are missing task evaluations for some systems. In summary, our method can be described in two steps:

### Our ranking procedure in a nutshell

1. **Matrix Representation of the rankings (Sssec. 2.2.1).** 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 ( $\sigma^l$ ) approach (see Sssec. 2.2.2) for both task-level and instance-level information and a two-level aggregation approach ( $\sigma^{2l}$ ) for instance-level information (see Sssec. 2.2.3).

### 2.2.1 Matrix representation of the rankings

**Intuition.** The first step in our algorithm is to summarize the available information in all tasks and to input the missing information in a consistent manner. To do this, we use a matrix representation  $M^\pi$  for each partial ranking  $\pi$ . 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 [53, 82, 83, 115], for computing distances among partial rankings [49], clustering [2] and classification [66] among others. However, these problems consider specific forms of missing data such as top- $k$  rankings [49] or bucket orderings [1]. Our approach differs from the aforementioned literature in the fact that we input the missing data in a consistent manner in order to be able to deal with arbitrary missing data.

**Efficiently building  $M^\pi$ .** Let us consider a partial ranking  $\pi$  and let  $M^\pi \in [0, 1]^{N \times N}$  be its matrix representation. Matrix  $M_{ij}^\pi$  denotes the proportion of complete rankings that are compatible with  $\pi$  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  $\pi$ .* In this case, we set  $M_{i,j}^\pi = 0$  if  $i \succ j$  else  $M_{i,j}^\pi = 1$
2. *if no information is provided for either system  $i$  or system  $j$  in  $\pi$ ,* meaning that both systems are unobserved in the partial ranking. In this case,  $M_{i,j}^\pi = 0.5$ , 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  $\pi$  (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. D).

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

### 2.2.2 Final System Ranking from Matrix Representation: a one level approach ( $\sigma^l$ )

**Intuition.** At this stage, we have demonstrated the construction of a matrix  $M^\pi$  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). Consequently, it becomes necessary to combine 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  $M^{\pi^{r_{t,k}}}$  is represent 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  $\sigma^l$  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 [\[31\]](#). Formally:

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

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

### 2.2.3 Final System Ranking from Matrix Representation: a two level approach ( $\sigma^{2l}$ )

**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  $\sigma^{2l}$  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  $\sigma^{2l,t} = \text{argsort} \left( \text{argsort} \left[ \sum_i M_{i,0}^t, \dots, \sum_i M_{i,N}^t \right] \right)$ .
3. Compute the Borda count aggregation  $\sigma^{2l}$  of  $[\sigma^{2l,1}, \dots, \sigma^{2l,t}, \dots, \sigma^{2l,T}]$ .

## 2.3 Confidence Intervals for $\sigma^l$

When evaluating 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 as Mallows Model [\[52\]](#) or Plackett-Luce [\[100\]](#),  $M_{ij}^\pi$  are random variables of known expected value. What we compute in the previous section is the empirical value of it,  $\widehat{M}_{ij}^\pi$  that approximates the true value  $M_{ij}^\pi$ . Here, we want to know how close these two quantities are. Formally, we are looking for a confidence interval of level  $\delta$ , that is the value for  $c_{ij}$  around  $\widehat{M}_{ij}^\pi$  that contains  $M_{ij}^\pi$  with high probability,  $P(|\widehat{M}_{ij}^\pi - M_{ij}^\pi| \geq c_{ij}) \leq 1 - \delta$ . Noting that  $0 \leq M_{ij}^\pi \leq 1$ ,



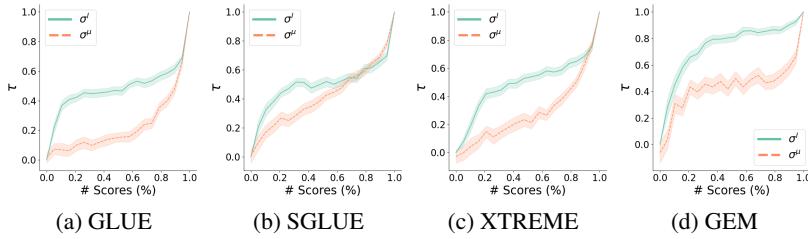


Figure 2: Task-Level Robustness Experiment. We compare the robustness of our method  $\sigma^l$  with the mean aggregation method  $\sigma^\mu$  by measuring the Kendall  $\tau$  correlation coefficient between their respective rankings after removing a proportion  $\eta$  of scores and by considering the whole scores.

we can use the Hoeffding inequality [63] to compute the value of the confidence interval:

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

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  $M_{ij}^\pi$  to 0.5. Thus,  $i$  performs better than  $j$  iff  $M_{ij}^\pi > .5$ . If the difference between  $M_{ij}^\pi$  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  $M_{ij}^\pi$  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 systems is better than the other with high probability. Similar approaches have been proposed to find complete rankings and best ranked systems with high probability [18, 124].

### 2.4 Baseline methods

To date, there is no established method for benchmarking NLP systems in the presence of 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 [98, 79, 84, 61, 95], and we will refer to it as  $\sigma^\mu$  in our experiments.

## 3 Synthetic Experiments

### 3.1 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  $\phi$  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  $\phi$  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  $[1, \dots, N]$ , with a noise term controlled by the dispersion parameter  $\phi$ . 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 for every data point.

### 3.2 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  $\lambda$ . Although this corruption

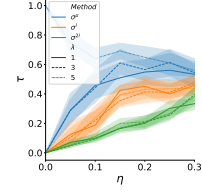


Figure 3: Rob. for missing data ( $\eta$ ) and different scaling corruptions ( $\lambda$ ).



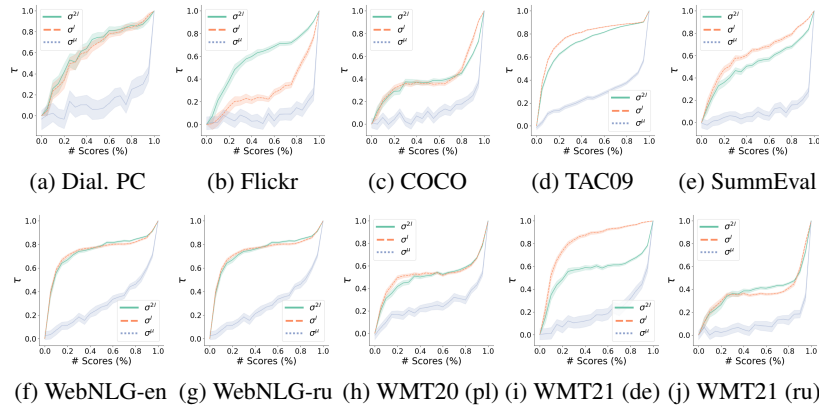


Figure 5: Instance-Level Robustness Experiment. We evaluate the robustness of our proposed aggregation methods, namely  $\sigma^{2l}$ ,  $\sigma^l$ , and the mean aggregation method  $\sigma^\mu$ , by randomly removing a proportion  $\eta$  of all instances on a specific task for a specific system.

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  $\lambda$  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.*

### 3.3 Pairwise Confidence Analysis

To determine the number of system comparisons required to achieve a desired confidence level of  $\delta$ , we use Eq. 4. Fig. 4 presents the results for two confidence levels ( $\delta$ ). 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  $\phi$ . As expected, when the rankings are more concentrated (i.e., when  $\phi$  is closer to 1), fewer system comparisons are needed to achieve a high number of valid system comparisons. In real-world benchmarks, test sets usually contain more than 500 pairs.

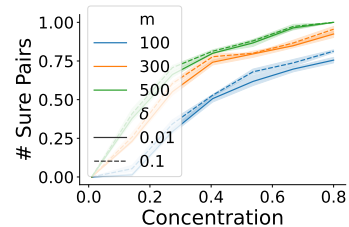


Figure 4: Confidence analysis.

## 4 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. 4.1 and Ap. B).

### 4.1 A Comprehensive Collection of NLP System Scores

Our dataset builds upon the one used in [31] 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 [126], SGLUE [125], and XTREME [64], which include tasks of varying natures such as accuracy, F1-score, and mean square errors. In addition, we collected data from the GEM Benchmark [56], which was an ideal use case for our methods as it encompasses missing data by design (as shown in Table 3 of [56]) 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 [4] 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. Firstly, we collected data from five distinct tasks - dialogue

[85], image description [133], summary evaluation [40, 92, 13, 48], data-to-text [55, 138], and translation [106]. For the translation part, we added datasets from WMT15 [123], WMT16 [15], WMT17 [14], WMT18 [110], WMT19 [9], WMT20 [81], and WMT21 [51] in several languages such as en, ru, ts, and others. Secondly, we expanded the set of used metrics from 10 to 17, including Rouge [77], JS [78], Bleu [94], Chrfpp [101], BERTScore [135], MoverScore [137], Baryscore [34], DepthScore [120], Infoml [29], CharErrorRate [86], ExtendedEditDistance [121], MatchErrorRate, TranslationEditRate [117], WordErrorRate [4], WordInfoLost [87], Bleurt [114], and Comet [107, 108]. 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.*

## 4.2 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 ( $\sigma^l$ ) against the mean aggregation method ( $\sigma^\mu$ ) and then we quantify the difference between the two output rankings.  $\sigma^l$  **is more robust than**  $\sigma^\mu$ . To compare the effectiveness of aggregation methods in handling missing values on real data, we randomly remove a proportion  $\eta$  of the task-level data and measure robustness by computing the Kendall  $\tau$  between the rankings of the systems obtained by considering the scores with and without missing values. From Fig. 2, we observe two extreme cases: when no systems are removed (*i.e.*,  $\eta = 0$ ), the aggregation methods output the same value as the one obtained with the full ranking and  $\tau = 1$ . At the other extreme, when all missing values are removed (*i.e.*,  $\eta = 1$ ), a total absence of correlation can be observed.

Overall, we find that  $\sigma^l$  achieves a higher correlation, with a large improvement of more than 10 points compared to other methods, especially in the medium corruption regime (*i.e.*,  $0.05 \geq \eta \geq 0.4$ ), which is more commonly encountered in practical scenarios. These results demonstrate that, on average, the rankings remain more stable when using our proposed method.

$\sigma^l$  **outputs a different ranking than**  $\sigma^\mu$ . We evaluated the correlation between different rankings obtained in the robustness experiment depicted in Fig. 2. Specifically, we compared the rankings produced by  $\sigma^l$  and  $\sigma^\mu$  in Tab. 1 and found a weak correlation between the two rankings, indicating that they produce different rankings. This is further supported by the results presented in Tab. 2,

which measure the percentage of times that the top 1 and top 3 rankings differ when considering the 2k rankings generated in the robustness experiment. *These results demonstrate that in addition to being more robust, our ranking procedure produces different conclusions when benchmarking systems in the presence of missing tasks.*

	$\tau_{\sigma^l \leftrightarrow \sigma^\mu}$	Dataset	top 1	top 3
GLUE	0.17 $\pm$ 0.24	GEM	0.52	0.25
SGLUE	0.33 $\pm$ 0.27	SGLUE	0.20	0.15
XTREM	0.26 $\pm$ 0.26	GLUE	0.10	0.07
GEM	0.36 $\pm$ 0.36	XTREM	0.19	0.09

Table 1: Agreement measured by Kendall  $\tau$  correlation.

Table 2: Percentage of times the top 1 and top 3 systems are the same between  $\sigma^l$  and  $\sigma^\mu$ .

## 4.3 Instance-Level Benchmarking in Real-World Scenarios

In this section, we evaluate the robustness of our proposed aggregation methods,  $\sigma^{2l}$ ,  $\sigma^l$ , and the baseline  $\sigma^\mu$ , in the presence of missing data. We also compare the rankings obtained from different algorithms on our large benchmark dataset, comprising over 132 million scores.

$\sigma^{2l}$  **and**  $\sigma^l$  **are more robust than**  $\sigma^\mu$ . Similarly to the previous robustness experiment, we randomly remove a proportion  $\eta$  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  $\tau$  correlation coefficient between the ranking obtained with missing values and the ranking obtained with complete scores.

Both  $\sigma^{2l}$  and  $\sigma^l$  produce highly correlated rankings, while being different from  $\sigma^\mu$ . We conducted a replication of the agreement analysis presented in Ssec. 4.3 and present the findings in Tab. 3 and Tab. 4. Our results align

	Corr.		Top 1	Top 3
$\tau_{\sigma^{2l} \leftrightarrow \sigma^l}$	$0.80 \pm 0.22$	$\sigma^{2l}$ vs $\sigma^l$	0.67	0.36
$\tau_{\sigma^l \leftrightarrow \sigma^\mu}$	$0.20 \pm 0.28$	$\sigma^l$ vs $\sigma^\mu$	0.21	0.09
$\tau_{\sigma^\mu \leftrightarrow \sigma^{2l}}$	$0.19 \pm 0.28$	$\sigma^\mu$ vs $\sigma^{2l}$	0.19	0.09

Table 3: Agreement.

Table 4: Top 1 and 3 analysis.

with those of our previous experiments, demonstrating that both of our ranking-based procedures ( $\sigma^{2l}$  and  $\sigma^l$ ) are more robust in the presence of missing data and yield different rankings than  $\sigma^\mu$ .

#### 4.4 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  $\sigma^l$  on WMT en-de with a confidence level of  $\delta = 0.1$ . Green value in position  $i < j$  illustrate that system  $0.5 \notin [\widehat{M}_{ij}^\pi - c_{ij}, \widehat{M}_{ij}^\pi + c_{ij}]$  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. These findings demonstrate that the confidence interval analysis provides insights into the relative performance of systems.

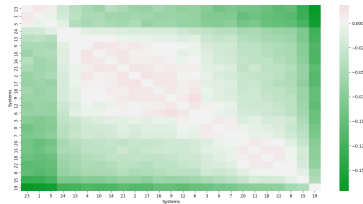


Figure 6: Confidence interval analysis on WMT en-de for a corruption level of  $\eta = 0.2$  and a confidence level  $\delta = 0.01$ . The final ranking can be seen on the x axis: left to right is best to worst

### 5 Conclusions and Future Research Directions

Our study sheds light on the limitations of the conventional mean-aggregation approach, particularly when dealing with missing data. To address this issue, we propose a novel statistical perspective and aggregation procedures that are both robust and grounded in social choice theory. We introduce two alternative methods: the one-level aggregation method ( $\sigma^l$ ) stands out as the most robust approach. Furthermore,  $\sigma^l$  allows to compute confidence intervals, enabling a more refined statistical analysis. By offering a more reliable and robust means of comparing different systems, our contribution equips practitioners, especially in the NLP field where large pre-trained models are expected to exhibit strong generalization across diverse tasks.

### 6 Acknowledgements

This work was also granted access to the HPC resources of IDRIS under the allocation 2021-AP010611665 as well as under the project 2021- 101838 made by GENCI.

### References

- [1] M. Achab, A. Korba, and S. Cl emen on. Dimensionality reduction and (bucket) ranking: a mass transportation approach. volume 98, pages 64–93. PMLR, 2019.
- [2] N. Ailon. Aggregation of partial rankings, p-ratings and top-m lists. *Algorithmica (New York)*, 2010.
- [3] A. Ali and M. Meil . Experiments with kemeny ranking: What works when? *Mathematical Social Sciences*, 64(1):28–40, 2012.
- [4] A. Ali and S. Renals. Word error rate estimation for speech recognition: e-wer. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 20–24, 2018.
- [5] V. Aribandi, Y. Tay, T. Schuster, J. Rao, H. S. Zheng, S. V. Mehta, H. Zhuang, V. Q. Tran, D. Bahri, J. Ni, et al. Ext5: Towards extreme multi-task scaling for transfer learning. *arXiv preprint arXiv:2111.10952*, 2021.

- [6] M. Artetxe, I. Aldabe, R. Agerri, O. Perez-de Viñaspre, and A. Soroa. 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, Dec. 2022. Association for Computational Linguistics.
- [7] M. Artetxe, S. Ruder, and D. Yogatama. On the cross-lingual transferability of monolingual representations. *arXiv preprint arXiv:1910.11856*, 2019.
- [8] M. Artetxe and H. Schwenk. Massively multilingual sentence embeddings for zero-shot cross-lingual transfer and beyond. *Transactions of the Association for Computational Linguistics*, 7:597–610, 2019.
- [9] L. Barrault, O. Bojar, M. R. Costa-Jussa, C. Federmann, M. Fishel, Y. Graham, B. Haddow, M. Huck, P. Koehn, S. Malmasi, et al. Findings of the 2019 conference on machine translation (wmt19). ACL, 2019.
- [10] J. J. Bartholdi, C. A. Tovey, and M. A. Trick. The computational difficulty of manipulating an election. *Social Choice and Welfare*, 6(3):227–241, 1989.
- [11] D. Bayer and P. Diaconis. Trailing the dovetail shuffle to its lair. *The Annals of Applied Probability*, 2:294–313, 1992.
- [12] L. Bentivogli, P. Clark, I. Dagan, and D. Giampiccolo. The fifth pascal recognizing textual entailment challenge. In *TAC*, 2009.
- [13] M. Bhandari, P. Gour, A. Ashfaq, P. Liu, and G. Neubig. Re-evaluating evaluation in text summarization. *arXiv preprint arXiv:2010.07100*, 2020.
- [14] O. Bojar, R. Chatterjee, C. Federmann, Y. Graham, B. Haddow, S. Huang, M. Huck, P. Koehn, Q. Liu, V. Logacheva, et al. Findings of the 2017 conference on machine translation (wmt17). Association for Computational Linguistics, 2017.
- [15] O. Bojar, R. Chatterjee, C. Federmann, Y. Graham, B. Haddow, M. Huck, A. J. Yepes, P. Koehn, V. Logacheva, C. Monz, et al. 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, 2016.
- [16] R. Bommasani, D. A. Hudson, E. Adeli, R. Altman, S. Arora, S. von Arx, M. S. Bernstein, J. Bohg, A. Bosselut, E. Brunskill, et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.
- [17] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.
- [18] R. Busa-Fekete, E. Hüllermeier, and B. Szörényi. 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, 2014.
- [19] D. Cer, M. Diab, E. Agirre, I. Lopez-Gazpio, and L. Specia. Semeval-2017 task 1: Semantic textual similarity-multilingual and cross-lingual focused evaluation. *arXiv preprint arXiv:1708.00055*, 2017.
- [20] E. Chapuis, P. Colombo, M. Labeau, and C. Clavel. Code-switched inspired losses for generic spoken dialog representations. *arXiv preprint arXiv:2108.12465*, 2021.
- [21] E. Chapuis, P. Colombo, M. Manica, M. Labeau, and C. Clavel. Hierarchical pre-training for sequence labelling in spoken dialog. *arXiv preprint arXiv:2009.11152*, 2020.
- [22] C. Chhun, P. Colombo, F. M. Suchanek, and C. Clavel. Of human criteria and automatic metrics: A benchmark of the evaluation of story generation. In N. Calzolari, C. Huang, H. Kim, J. Pustejovsky, L. Wanner, K. Choi, P. Ryu, H. Chen, L. Donatelli, H. Ji, S. Kurohashi, P. Paggio, N. Xue, S. Kim, Y. Hahm, Z. He, T. K. Lee, E. Santus, F. Bond, and S. Na, editors, *Proceedings of the 29th International Conference on Computational Linguistics, COLING 2022, Gyeongju, Republic of Korea, October 12-17, 2022*, pages 5794–5836. International Committee on Computational Linguistics, 2022.

- [23] C. Clark, K. Lee, M.-W. Chang, T. Kwiatkowski, M. Collins, and K. Toutanova. Boolq: Exploring the surprising difficulty of natural yes/no questions. *arXiv preprint arXiv:1905.10044*, 2019.
- [24] J. H. Clark, E. Choi, M. Collins, D. Garrette, T. Kwiatkowski, V. Nikolaev, and J. Palomaki. Tydi qa: A benchmark for information-seeking question answering in typologically diverse languages. *Transactions of the Association for Computational Linguistics*, 8:454–470, 2020.
- [25] S. Cléménçon, R. Gaudel, and J. Jakubowicz. 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, 2011.
- [26] P. Colombo. *Apprendre à représenter et à générer du texte en utilisant des mesures d’information. (Learning to represent and generate text using information measures)*. PhD thesis, Polytechnic Institute of Paris, France, 2021.
- [27] P. Colombo. *Learning to represent and generate text using information measures*. PhD thesis, Institut polytechnique de Paris, 2021.
- [28] P. Colombo, E. Chapuis, M. Manica, E. Vignon, G. Varni, and C. Clavel. Guiding attention in sequence-to-sequence models for dialogue act prediction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 7594–7601, 2020.
- [29] P. Colombo, C. Clave, and P. Piantanida. Infoml: A new metric to evaluate summarization & data2text generation. *arXiv preprint arXiv:2112.01589*, 2021.
- [30] P. Colombo, C. Clavel, and P. Piantanida. A novel estimator of mutual information for learning to disentangle textual representations. *arXiv preprint arXiv:2105.02685*, 2021.
- [31] P. Colombo, N. Noiry, E. Irurozki, and S. CLEMENCON. What are the best systems? new perspectives on NLP benchmarking. In A. H. Oh, A. Agarwal, D. Belgrave, and K. Cho, editors, *Advances in Neural Information Processing Systems*, 2022.
- [32] P. Colombo, N. Noiry, E. Irurozki, and S. Cléménçon. What are the best systems? new perspectives on NLP benchmarking. In *NeurIPS*, 2022.
- [33] P. Colombo, M. Peyrard, N. Noiry, R. West, and P. Piantanida. The glass ceiling of automatic evaluation in natural language generation. *arXiv preprint arXiv:2208.14585*, 2022.
- [34] P. Colombo, G. Staerman, C. Clavel, and P. Piantanida. Automatic text evaluation through the lens of wasserstein barycenters. *arXiv preprint arXiv:2108.12463*, 2021.
- [35] P. Colombo, G. Staerman, N. Noiry, and P. Piantanida. Learning disentangled textual representations via statistical measures of similarity. In S. Muresan, P. Nakov, and A. Villavicencio, editors, *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 2614–2630. Association for Computational Linguistics, 2022.
- [36] P. Colombo, W. Witon, A. Modi, J. Kennedy, and M. Kapadia. Affect-driven dialog generation. *arXiv preprint arXiv:1904.02793*, 2019.
- [37] P. Colombo, C. Yang, G. Varni, and C. Clavel. Beam search with bidirectional strategies for neural response generation. *arXiv preprint arXiv:2110.03389*, 2021.
- [38] A. Conneau, G. Lample, R. Rinott, A. Williams, S. R. Bowman, H. Schwenk, and V. Stoyanov. Xnli: Evaluating cross-lingual sentence representations. *arXiv preprint arXiv:1809.05053*, 2018.
- [39] I. Dagan, O. Glickman, and B. Magnini. The pascal recognising textual entailment challenge. In *Machine Learning Challenges Workshop*, pages 177–190. Springer, 2005.
- [40] H. T. Dang, K. Owczarzak, et al. Overview of the tac 2008 update summarization task. In *TAC*, 2008.

- [41] M. Darrin, P. Piantanida, and P. Colombo. Rainproof: An umbrella to shield text generators from out-of-distribution data. *CoRR*, abs/2212.09171, 2022.
- [42] M. Darrin, G. Staerman, E. D. C. Gomes, J. C. Cheung, P. Piantanida, and P. Colombo. Unsupervised layer-wise score aggregation for textual OOD detection. *CoRR*, abs/2302.09852, 2023.
- [43] M.-C. De Marneffe, M. Simons, and J. Tonhauser. The commitmentbank: Investigating projection in naturally occurring discourse. In *proceedings of Sinn und Bedeutung*, volume 23, pages 107–124, 2019.
- [44] M. Dehghani, Y. Tay, A. A. Gritsenko, Z. Zhao, N. Houlsby, F. Diaz, D. Metzler, and O. Vinyals. The benchmark lottery. *arXiv preprint arXiv:2107.07002*, 2021.
- [45] T. Dinkar, P. Colombo, M. Labeau, and C. Clavel. The importance of fillers for text representations of speech transcripts. *arXiv preprint arXiv:2009.11340*, 2020.
- [46] W. B. Dolan and C. Brockett. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*, 2005.
- [47] C. Dwork, R. Kumar, M. Naor, and D. Sivakumar. Rank aggregation methods for the web. In *Proceedings of the 10th international conference on World Wide Web*, pages 613–622, 2001.
- [48] A. R. Fabbri, W. Kryściński, B. McCann, C. Xiong, R. Socher, and D. Radev. Summeval: Re-evaluating summarization evaluation. *Transactions of the Association for Computational Linguistics*, 9:391–409, 2021.
- [49] R. Fagin, R. Kumar, and D. Sivakumar. Comparing top k lists. *SIAM Journal on Discrete Mathematics*, 2003.
- [50] A. Fan, S. Bhosale, H. Schwenk, Z. Ma, A. El-Kishky, S. Goyal, M. Baines, O. Celebi, G. Wenzek, V. Chaudhary, et al. Beyond english-centric multilingual machine translation. *The Journal of Machine Learning Research*, 22(1):4839–4886, 2021.
- [51] A. Farhad, A. Arkady, B. Magdalena, B. Ondřej, C. Rajen, C. Vishrav, M. R. Costa-jussa, E.-B. Cristina, F. Angela, F. Christian, et al. 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, 2021.
- [52] M. A. Fligner and J. S. Verducci. Distance based ranking models. *Journal of the Royal Statistical Society*, 48(3):359–369, 1986.
- [53] J. Fürnkranz and E. Hüllermeier. Pairwise preference learning and ranking. pages 145–156, 2003.
- [54] A. Garcia, P. Colombo, S. Essid, F. d’Alché Buc, and C. Clavel. From the token to the review: A hierarchical multimodal approach to opinion mining. *arXiv preprint arXiv:1908.11216*, 2019.
- [55] C. Gardent, A. Shimorina, S. Narayan, and L. Perez-Beltrachini. The webnlg challenge: Generating text from rdf data. In *Proceedings of the 10th International Conference on Natural Language Generation*, pages 124–133, 2017.
- [56] S. Gehrmann, T. Adewumi, K. Aggarwal, P. S. Ammanamanchi, A. Anuoluwapo, A. Bosselut, K. R. Chandu, M. Clinciu, D. Das, K. D. Dhole, et al. The gem benchmark: Natural language generation, its evaluation and metrics. *arXiv preprint arXiv:2102.01672*, 2021.
- [57] S. Gehrmann, A. Bhattacharjee, A. Mahendiran, A. Wang, A. Papangelis, A. Madaan, A. McMillan-Major, A. Shvets, A. Upadhyay, B. Yao, et al. Gemv2: Multilingual nlg benchmarking in a single line of code. *arXiv preprint arXiv:2206.11249*, 2022.
- [58] S. Gehrmann, E. Clark, and T. Sellam. Repairing the cracked foundation: A survey of obstacles in evaluation practices for generated text. *arXiv preprint arXiv:2202.06935*, 2022.

- [59] I. M. Gessel and Y. Zhuang. Shuffle-compatible permutation statistics. *Advances in Mathematics*, 332:85–141, 2018.
- [60] D. Giampiccolo, B. Magnini, I. Dagan, and W. B. Dolan. The third pascal recognizing textual entailment challenge. In *Proceedings of the ACL-PASCAL workshop on textual entailment and paraphrasing*, pages 1–9, 2007.
- [61] G. Guibon, M. Labeau, H. Flamein, L. Lefeuvre, and C. Clavel. 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, Nov. 2021. Association for Computational Linguistics.
- [62] R. Herbrich, T. Minka, and T. Graepel. Trueskill™: A bayesian skill rating system. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems*, volume 19. MIT Press, 2006.
- [63] W. Hoeffding. Probability inequalities for sums of bounded random variables. *The collected works of Wassily Hoeffding*, pages 409–426, 1994.
- [64] J. Hu, S. Ruder, A. Siddhant, G. Neubig, O. Firat, and M. Johnson. Xtreme: A massively multilingual multi-task benchmark for evaluating cross-lingual generalisation. In *International Conference on Machine Learning*, pages 4411–4421. PMLR, 2020.
- [65] D. Hupkes, M. Giulianelli, V. Dankers, M. Artetxe, Y. Elazar, T. Pimentel, C. Christodoulopoulos, K. Lasri, N. Saphra, A. Sinclair, et al. State-of-the-art generalisation research in nlp: a taxonomy and review. *arXiv preprint arXiv:2210.03050*, 2022.
- [66] E. Hüllermeier, J. Fürnkranz, W. Cheng, and K. Brinker. Label ranking by learning pairwise preferences. *Artificial Intelligence*, 172:1897–1916, 2008.
- [67] H. Jalalzai, P. Colombo, C. Clavel, É. Gaussier, G. Varni, E. Vignon, and A. Sabourin. Heavy-tailed representations, text polarity classification & data augmentation. *Advances in Neural Information Processing Systems*, 33:4295–4307, 2020.
- [68] D. Khashabi, S. Chaturvedi, M. Roth, S. Upadhyay, and D. Roth. 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, 2018.
- [69] D. E. Knuth. Permutations, matrices and generalized young tableaux. *Pacific Journal of Mathematics*, 34:709–727, 1970.
- [70] J. Kocoń, I. Cichecki, O. Kaszyca, M. Kochanek, D. Szydło, J. Baran, J. Bielaniewicz, M. Gruza, A. Janz, K. Kanclerz, A. Kocoń, B. Koptyra, W. Mieleszczenko-Kowszewicz, P. Miłkowski, M. Oleksy, M. Piasecki, Łukasz Radliński, K. Wojtasik, S. Woźniak, and P. Kazienko. Chatgpt: Jack of all trades, master of none, 2023.
- [71] J. Kocoń, I. Cichecki, O. Kaszyca, M. Kochanek, D. Szydło, J. Baran, J. Bielaniewicz, M. Gruza, A. Janz, K. Kanclerz, A. Kocoń, B. Koptyra, W. Mieleszczenko-Kowszewicz, P. Miłkowski, M. Oleksy, M. Piasecki, Łukasz Radliński, K. Wojtasik, S. Woźniak, and P. Kazienko. Chatgpt: Jack of all trades, master of none, 2023.
- [72] R. Kondor and M. Barbosa. Ranking with kernels in fourier space. 2010.
- [73] R. Kondor and W. Dempsey. Multiresolution analysis on the symmetric group. *Advances in Neural Information Processing Systems*, 25, 2012.
- [74] E. Lehman, E. Hernandez, D. Mahajan, J. Wulff, M. J. Smith, Z. Ziegler, D. Nadler, P. Szolovits, A. Johnson, and E. Alsentzer. Do we still need clinical language models?, 2023.
- [75] H. Levesque, E. Davis, and L. Morgenstern. The winograd schema challenge. In *Thirteenth International Conference on the Principles of Knowledge Representation and Reasoning*, 2012.



- [76] P. Lewis, B. Oğuz, R. Rinott, S. Riedel, and H. Schwenk. Mlqa: Evaluating cross-lingual extractive question answering. *arXiv preprint arXiv:1910.07475*, 2019.
- [77] C.-Y. Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81, 2004.
- [78] C.-Y. Lin, G. Cao, J. Gao, and J.-Y. Nie. 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, 2006.
- [79] X. V. Lin, T. Mihaylov, M. Artetxe, T. Wang, S. Chen, D. Simig, M. Ott, N. Goyal, S. Bhosale, J. Du, R. Pasunuru, S. Shleifer, P. S. Koura, V. Chaudhary, B. O’Horo, J. Wang, L. Zettlemoyer, Z. Kozareva, M. Diab, V. Stoyanov, and X. Li. 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, Dec. 2022. Association for Computational Linguistics.
- [80] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.
- [81] B. Loïc, B. Magdalena, B. Ondřej, F. Christian, G. Yvette, G. Roman, H. Barry, H. Matthias, J. Eric, K. Tom, et al. 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,, 2020.
- [82] T. Lu and C. Boutilier. Effective sampling and learning for mallows models with pairwise-preference data. *Journal of Machine Learning Research*, 2014.
- [83] T. Lu and C. Boutilier. Effective sampling and learning for mallows models with pairwise-preference data. *Journal of Machine Learning Research*, 2014.
- [84] L. Martin, B. Muller, P. J. Ortiz Suárez, Y. Dupont, L. Romary, É. de la Clergerie, D. Seddah, and B. Sagot. CamemBERT: a tasty French language model. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7203–7219, Online, July 2020. Association for Computational Linguistics.
- [85] S. Mehri and M. Eskenazi. Usr: An unsupervised and reference free evaluation metric for dialog generation. *arXiv preprint arXiv:2005.00456*, 2020.
- [86] A. Morris, V. Maier, and P. Green. From wer and ril to mer and wil: improved evaluation measures for connected speech recognition. 01 2004.
- [87] A. C. Morris, V. Maier, and P. Green. From wer and ril to mer and wil: improved evaluation measures for connected speech recognition. In *Eighth International Conference on Spoken Language Processing*, 2004.
- [88] J.-P. Ng and V. Abrecht. Better summarization evaluation with word embeddings for rouge. *arXiv preprint arXiv:1508.06034*, 2015.
- [89] J. Nivre, M. Abrams, Z. Agic, L. Ahrenberg, L. Antonsen, et al. 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>, 2018.
- [90] J. Novikova, O. Dušek, and V. Rieser. Rankme: Reliable human ratings for natural language generation. *arXiv preprint arXiv:1803.05928*, 2018.
- [91] OpenAI. Gpt-4 technical report, 2023.
- [92] K. Owczarzak and H. T. Dang. 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, 2011.

- [93] X. Pan, B. Zhang, J. May, J. Nothman, K. Knight, and H. Ji. 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, 2017.
- [94] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu. 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, 2002.
- [95] Y. Peng, S. Yan, and Z. Lu. 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, Aug. 2019. Association for Computational Linguistics.
- [96] M. Peyrard, T. Botschen, and I. Gurevych. Learning to score system summaries for better content selection evaluation. In *Proceedings of the Workshop on New Frontiers in Summarization*, pages 74–84, 2017.
- [97] M. Peyrard, W. Zhao, S. Eger, and R. West. Better than average: Paired evaluation of nlp systems. *arXiv preprint arXiv:2110.10746*, 2021.
- [98] J. Pfeiffer, N. Goyal, X. Lin, X. Li, J. Cross, S. Riedel, and M. Artetxe. 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, July 2022. Association for Computational Linguistics.
- [99] M. T. Pilehvar and J. Camacho-Collados. Wic: the word-in-context dataset for evaluating context-sensitive meaning representations. *arXiv preprint arXiv:1808.09121*, 2018.
- [100] R. L. Plackett. The analysis of permutations. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 24(2):193–202, 1975.
- [101] M. Popović. chrF++: words helping character n-grams. In *Proceedings of the second conference on machine translation*, pages 612–618, 2017.
- [102] M. Post. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Belgium, Brussels, Oct. 2018. Association for Computational Linguistics.
- [103] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551, 2020.
- [104] A. Rahimi, Y. Li, and T. Cohn. Massively multilingual transfer for ner. *arXiv preprint arXiv:1902.00193*, 2019.
- [105] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*, 2016.
- [106] T. Ranasinghe, C. Orasan, and R. Mitkov. An exploratory analysis of multilingual word-level quality estimation with cross-lingual transformers. *arXiv preprint arXiv:2106.00143*, 2021.
- [107] R. Rei, J. G. de Souza, D. Alves, C. Zerva, A. C. Farinha, T. Glushkova, A. Lavie, L. Coheur, and A. F. Martins. Comet-22: Unbabel-ist 2022 submission for the metrics shared task. In *Proceedings of the Seventh Conference on Machine Translation (WMT)*, pages 578–585, 2022.
- [108] R. Rei, C. Stewart, A. C. Farinha, and A. Lavie. Comet: A neural framework for mt evaluation. *arXiv preprint arXiv:2009.09025*, 2020.
- [109] M. Reid and M. Artetxe. 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, July 2022. Association for Computational Linguistics.

- [110] O. re j Bojar, C. Federmann, M. Fishel, Y. Graham, B. Haddow, M. Huck, P. Koehn, and C. Monz. Findings of the 2018 conference on machine translation (wmt18). In *Proceedings of the Third Conference on Machine Translation*, volume 2, pages 272–307, 2018.
- [111] M. Roemmele, C. A. Bejan, and A. S. Gordon. Choice of plausible alternatives: An evaluation of commonsense causal reasoning. In *2011 AAAI Spring Symposium Series*, 2011.
- [112] S. Ruder. Challenges and Opportunities in NLP Benchmarking. <http://ruder.io/nlp-benchmarking>, 2021.
- [113] J. Sedoc and L. Ungar. 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, Nov. 2020. Association for Computational Linguistics.
- [114] T. Sellam, D. Das, and A. P. Parikh. Bleurt: Learning robust metrics for text generation. *arXiv preprint arXiv:2004.04696*, 2020.
- [115] N. B. Shah, S. Balakrishnan, A. Guntuboyina, and M. J. Wainwright. Stochastically transitive models for pairwise comparisons: Statistical and computational issues. *IEEE Transactions on Information Theory*, 2017.
- [116] E. Sibony, S. Cl  men  on, and J. Jakubowicz. Mra-based statistical learning from incomplete rankings. In *International Conference on Machine Learning*, pages 1432–1441. PMLR, 2015.
- [117] M. Snover, B. Dorr, R. Schwartz, L. Micciulla, and J. Makhoul. A study of translation edit rate with targeted human annotation. In *Proceedings of association for machine translation in the Americas*, number 6. Cambridge, MA, 2006.
- [118] R. Socher, A. Perelygin, J. Wu, J. Chuang, C. D. Manning, A. Y. Ng, and C. Potts. 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, 2013.
- [119] A. Srivastava, A. Rastogi, A. Rao, A. A. M. Shueb, A. Abid, A. Fisch, A. R. Brown, A. Santoro, A. Gupta, A. Garriga-Alonso, et al. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *arXiv preprint arXiv:2206.04615*, 2022.
- [120] G. Staerman, P. Mozharovskiy, S. Cl  men  on, and F. d’Alch   Buc. Depth-based pseudo-metrics between probability distributions. *arXiv e-prints*, pages arXiv–2103, 2021.
- [121] P. Stanchev, W. Wang, and H. Ney. EED: Extended edit distance measure for machine translation. In *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, pages 514–520, Florence, Italy, Aug. 2019. Association for Computational Linguistics.
- [122] R. P. Stanley. *Enumerative Combinatorics*. Wadsworth Publishing Company, 1986.
- [123] M. Stanojevi  , A. Kamran, P. Koehn, and O. Bojar. Results of the wmt15 metrics shared task. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 256–273, 2015.
- [124] B. Sz  r  nyi, R. Busa-Fekete, A. Paul, and E. H  llermeier. Online rank elicitation for plackett-luce: A dueling bandits approach. *Advances in Neural Information Processing Systems*, 28, 2015.
- [125] A. Wang, Y. Pruksachatkun, N. Nangia, A. Singh, J. Michael, F. Hill, O. Levy, and S. R. Bowman. Superglue: A stickier benchmark for general-purpose language understanding systems. *arXiv preprint arXiv:1905.00537*, 2019.
- [126] A. Wang, A. Singh, J. Michael, F. Hill, O. Levy, and S. R. Bowman. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*, 2018.
- [127] A. Warstadt, A. Singh, and S. R. Bowman. Neural network acceptability judgments. *Transactions of the Association for Computational Linguistics*, 7:625–641, 2019.

- [128] H. S. Wilf. *East Side, West Side . . . - an introduction to combinatorial families-with Maple programming*. arxiv, 1999.
- [129] A. Williams, N. Nangia, and S. R. Bowman. A broad-coverage challenge corpus for sentence understanding through inference. *arXiv preprint arXiv:1704.05426*, 2017.
- [130] A. Williams, N. Nangia, and S. R. Bowman. A broad-coverage challenge corpus for sentence understanding through inference. *arXiv preprint arXiv:1704.05426*, 2017.
- [131] W. Witon, P. Colombo, A. Modi, and M. Kapadia. Disney at IEST 2018: Predicting emotions using an ensemble. In A. Balahur, S. M. Mohammad, V. Hoste, and R. Klinger, editors, *Proceedings of the 9th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis, WASSA@EMNLP 2018, Brussels, Belgium, October 31, 2018*, pages 248–253. Association for Computational Linguistics, 2018.
- [132] Y. Yang, Y. Zhang, C. Tar, and J. Baldridge. Paws-x: A cross-lingual adversarial dataset for paraphrase identification. *arXiv preprint arXiv:1908.11828*, 2019.
- [133] P. Young, A. Lai, M. Hodosh, and J. Hockenmaier. 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, 2014.
- [134] S. Zhang, X. Liu, J. Liu, J. Gao, K. Duh, and B. Van Durme. Record: Bridging the gap between human and machine commonsense reading comprehension. *arXiv preprint arXiv:1810.12885*, 2018.
- [135] T. Zhang, V. Kishore, F. Wu, K. Q. Weinberger, and Y. Artzi. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*, 2019.
- [136] Y. Zhang, J. Baldridge, and L. He. Paws: Paraphrase adversaries from word scrambling. *arXiv preprint arXiv:1904.01130*, 2019.
- [137] W. Zhao, M. Peyrard, F. Liu, Y. Gao, C. M. Meyer, and S. Eger. Moverscore: Text generation evaluating with contextualized embeddings and earth mover distance. *arXiv preprint arXiv:1909.02622*, 2019.
- [138] G. Zhou and G. Lampouras. 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, 2020.
- [139] P. Zweigenbaum, S. Sharoff, and R. Rapp. 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, 2017.
- [140] P. Zweigenbaum, S. Sharoff, and R. Rapp. 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, 2018.

# Appendices

<b>A Ethical Statement &amp; Limitation of our work</b>	<b>20</b>
<b>B Dataset Description</b>	<b>20</b>
B.1 Task Level Information . . . . .	20
B.2 Instance Level Information . . . . .	20
B.3 Data Statistics . . . . .	20
<b>C Additional Real-Data Experiments</b>	<b>21</b>
C.1 Example of Ranking with missing data on XTREM . . . . .	21
C.2 Additional Robustness Experiment on instance level datasets . . . . .	21
C.3 Additional Confidence Analysis on Task Level . . . . .	23
C.4 Future Work . . . . .	24
<b>D On the Rankings</b>	<b>24</b>
D.1 Borda Count on permutations (in vector notation) . . . . .	24
D.2 Borda Count on permutations in pairwise matrix notation . . . . .	24
D.3 Generating all compatible rankings . . . . .	25
D.4 Proof of uniformity . . . . .	25

## A 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.

## B Dataset Description

### B.1 Task Level Information

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

We gathered data from four benchmark studies, namely GLUE (General Language Understanding Evaluation) [126], SGLUE (SuperGLUE) [125]<sup>2</sup>, XTREME [64] 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 [127, 118, 46, 19, 105, 129, 39, 60, 12, 75]. 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 [23, 43, 111, 68, 134, 75, 99]. 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) [38, 130, 132, 136, 89, 104, 93, 140, 139, 8, 7, 105, 76, 24].

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.

### B.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 [40], TAC10, TAC11 [92], RSUM [13], and SEVAL [48] datasets. Regarding *sentence-based image description*, we rely on the FLICKR dataset [133]. For *dialogue*, we make use of the PersonaChat (PC) and TopicalChat (TC) datasets [85]. For the translation part, we added datasets from WMT15 [123], WMT16 [15], WMT17 [14], WMT18 [110], WMT19 [9], WMT20 [81], and WMT21 [51] 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) [96], ROUGE [77] (including five of its variants [88]), JS [1-2] [78], ChrFpp [101], BLEU, BERTScore [135], and MoverScore [137]. For the MLQE dataset, we solely consider several versions of BERTScore, MoverScore, and ContrastScore. Additionally, we incorporate human evaluation, which is specific to each dataset.

### B.3 Data Statistics

To give to the reader a better sense of the richness of our benchmark, we report in Fig. 7 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. 8 and Fig. 9, the smaller datasets consist of several hundred sentences in total, while the larger datasets encompass up to several hundred thousand sentences in total.

---

<sup>2</sup>Results can be accessed at <https://super.gluebenchmark.com/>

## C 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.

### C.1 Example of Ranking with missing data on XTREM

In this section, we aim to illustrate the distinction between different rankings obtained using  $\sigma^l$  and  $\sigma^\mu$  on XTREM dataset for a specific noise realisation. Using [Tab. 5](#), we obtaine the following rankings:

- $\sigma^l$  gives the following ranking :  $M0 > M3 > M2 > M1 > M7 > M5 > M4 > M8 > M9$
- $\sigma^\mu$  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 with our experiements, the ranking based method is more robust.

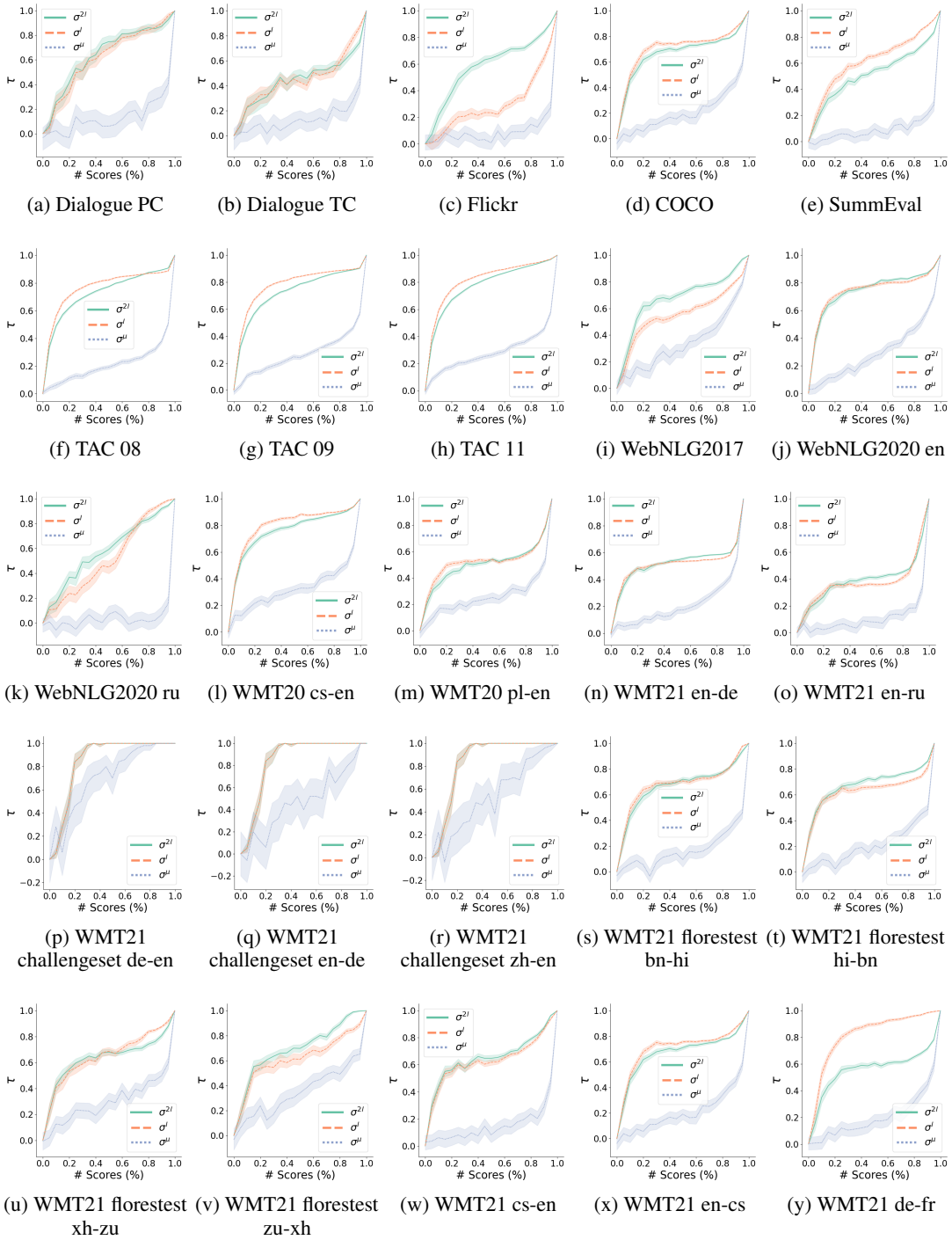
Model	Classification	Structured Prediction	Question Answering	Sentence Retrieval
M0	90.3	X	76.3	93.7
M1	90.1	X	75.0	X
M2	89.3	75.5	75.2	92.4
M3	89.0	76.7	73.4	93.3
M4	88.3	X	X	X
M5	X	X	X	X
M6	87.9	75.6	X	91.9
M7	X	X	X	92.6
M8	X	75.4	X	X
M9	88.2	74.6	X	89.0

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

### C.2 Additional Robustness Experiment on instance level datasets

In this section we report additionnal experiements on the instance level robustness.





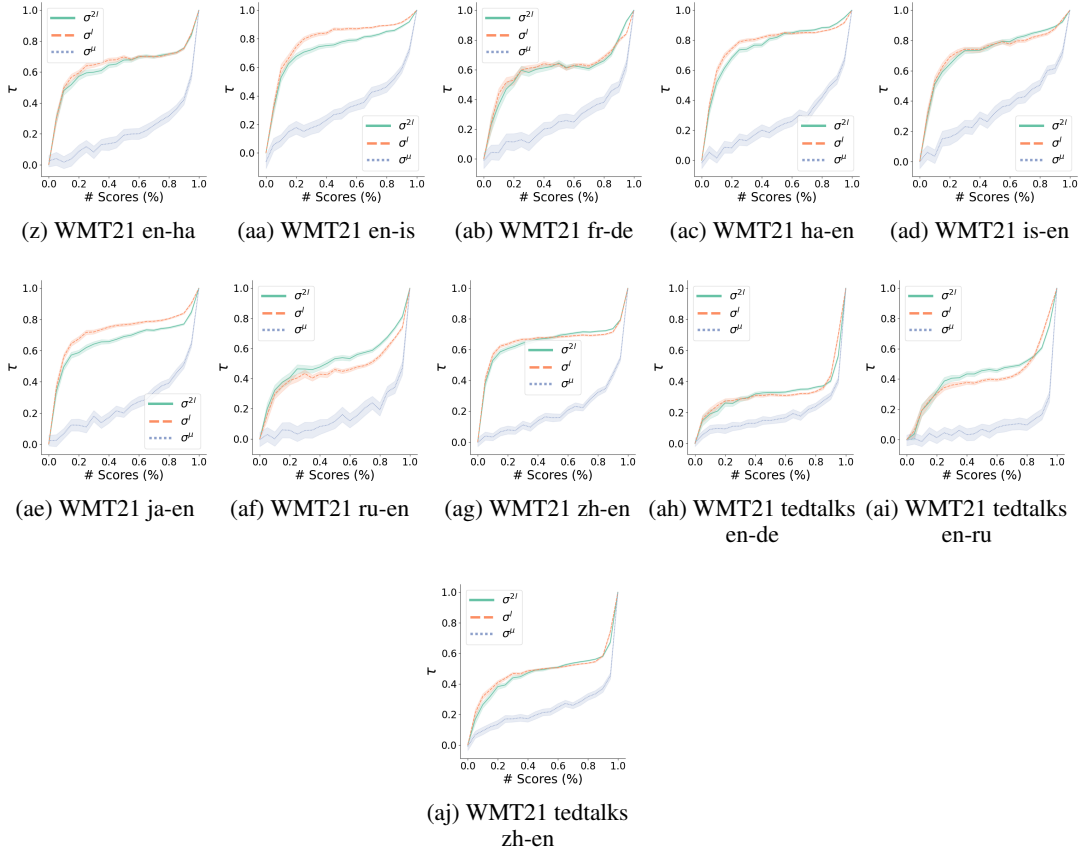


Figure 10: Instance-Level Robustness Experiment. We evaluate the robustness of our proposed aggregation methods, namely  $\sigma^{2l}$ ,  $\sigma^l$ , and the mean aggregation method  $\sigma^\mu$ , by randomly removing a proportion  $\eta$  of all instances on a specific task for a specific system. Each experiment is repeated 100 times for each proportion.

### C.3 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 Ssec. 4.4. 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.

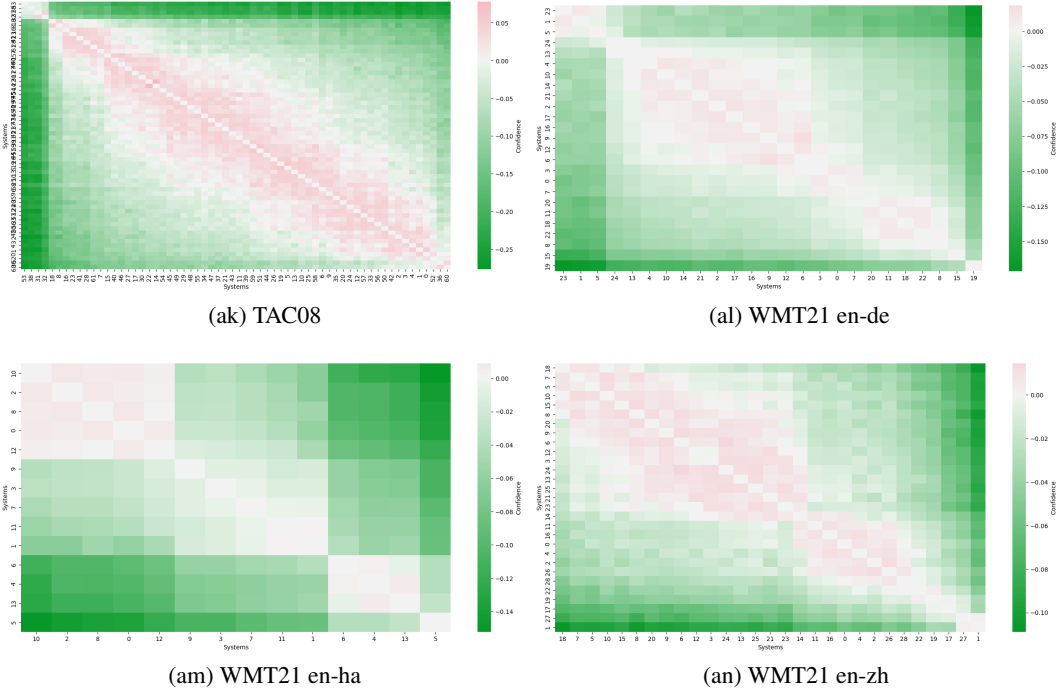


Figure 11: Confidence intervals for various instance level datasets with  $\eta = 0.2$  and  $\delta = 0.01$

## C.4 Future Work

In the futur we would like to extend our work to other sequence generation task [36, 45, 67, 21, 37], fairness [27, 30], safe ai [26, 22, 32, 35, 42, 41], classification [131, 54, 28, 20]

## D On the Rankings

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

### D.1 Borda Count on permutations (in vector notation)

**Remark 2.** *The Borda count is a ranking system that aggregates a set of permutations  $\sigma^1, \dots, \sigma^L \in \mathfrak{S}_N$  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  $\sigma := \text{Borda}(\sigma^1, \dots, \sigma^L) \in \mathfrak{S}_N$  that ranks the sums,  $\text{sum}_n$  ( $\text{argsort}(\text{argsort}(\text{sum}_1, \dots, \text{sum}_T))$ ).

### D.2 Borda Count on permutations in pairwise matrix notation

In [Sssec. 2.2.1](#) we argue that a ranking  $\sigma \in \mathfrak{S}_N$  can also be written as a pairwise matrix and in [Sssec. 2.2.2](#) and [Sssec. 2.2.3](#) 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  $\sigma$  for the Borda

count algorithm can be shown to be equivalent to the permutation that sorts the sum of the columns in  $M^D$ ,

$$\sigma = \text{argsort} \left( \text{argsort} \left[ \sum_i M_{i,0}^D, \dots, \sum_i M_{i,N}^D \right] \right). \quad (5)$$

### D.3 Generating all compatible rankings

In this section, we detail the computation of the  $M_{i,j}^\pi$  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  $\pi$ , item  $i$  is not evaluated, item  $j$  is evaluated and  $r$  is the (partial) rank of item  $j$ . Under this setting, we set  $M_{i,j}^\pi = p(n, k, r)$ , i.e., the proportion of compatible rankings that rank  $i$  before  $j$  when  $\pi$  has  $k$  items. The closed-form expressions for this quantities is given in Eq. 6. Here we note that  $t(n, k)$  is the total number rankings of  $n$  items of compatible with  $\pi$ ,  $S_b^a$  is the number of shuffles of two lists of lengths  $a$  and  $b$  and  $V_b^a$  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.

$$\begin{aligned} p(n, k, r) &= \sum_{i=0}^{n-k-1} V_{n-k-1}^i * (i+1) * S_{i+1}^r (n-k-i-1)! * S_{k-r-1}^{n-k-i-1} / t(n, k) \\ t(n, k) &= (n-k)! * S_{n-k}^k \\ S_b^a &= (a+b)! / (a! + b!) \\ V_b^a &= a! / (b-a)! \end{aligned} \quad (6)$$

**Remark 3.** A naive algorithm for generating the matrix  $M^\pi$  from  $\sigma \in S_{N-r,k}$  would have factorial complexity and it is thus exorbitant in practice for 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  $M^\pi$  from partial ranking  $\pi$ .

### D.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 [122, 128]: 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 [69].

**Shuffle two lists - *shuffle*( $A, B$ )** Given two disjoint lists of distinct elements  $A, B$  of lengths  $a, b$  respectively, generate a permutation  $\sigma$  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  $\sigma$ . This name and idea is based on the popular way of shuffling two decks of cards [11]. Its easy to see that Algorithm 1 generates every possible shuffling with equal probability. The total number of shuffles of lists  $A, B$  is given in Eq. 6 as  $S_b^a$ .

---

**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

```

---

**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 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  $\beta$  be a partial ranking of length  $k$  which includes item  $j$  in rank  $r$ ,  $\beta_1 \succ \dots \succ \beta_r = j \succ \dots \succ \beta_k$ . Let  $\eta$  be a disjoint set of  $n - k$  items which have not been ranked and which includes the unobserved item  $i$ . The goal is to generate (i) a compatible ranking with  $\beta$  (a ranking  $\sigma$  of all the items in such a way that the relative ordering of the items of  $\beta$  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 2 we select  $s - 1$  items randomly from  $\eta$ , where order of the items matter (i.e., a variation). Then, we insert item  $i$  in a random position of this list, denoted  $\eta_{head}$  in line 3. In line 4 we shuffle these two lists, i.e.,  $\eta_{head}$  and the  $r$ -head of  $\beta$ ,  $\beta_{head}$ , 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  $\sigma$ . We permute the rest of the unobserved items denoting these list  $\eta_{tail}$ , in line 6. Finally, we shuffle this list  $\eta_{tail}$  and the  $k - r$ -tail of  $\eta$  in line 7. The result of this shuffle is the tail of  $\sigma$ . Finally, in line 8 we return the concatenation of  $\sigma_{head, j}, \sigma_{tail}$ , which is clearly a compatible permutation with  $\beta$  as the relative order of the items in  $\beta$  is maintained in the output.

---

**Algorithm 2:** Generate a random ranking among those compatible with  $\beta$

---

```

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

```

---

It is easy to see that Algorithm 2 generates the target permutations uniformly at random. Following a classic strategy on Enumerative Combinatorics [122, 128] 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  $\beta$  that rank the unobserved item  $i$  before the observed item  $j$  is given by the following expression,*

$$p(n, k, r) = \sum_{i=0}^{n-k-1} V_{n-k-1}^i * (i + 1) * S_{i+1}^r (n - k - i - 1)! * S_{k-r-1}^{n-k-i-1} / t(n, k).$$

*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  $\beta$  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  $\eta_{head}$  and  $\beta_{head}$ .
- Line 6: There are  $(n - k - s - 1)!$  possible permutations of the items in  $\eta_{tail}$ .

- 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.

□

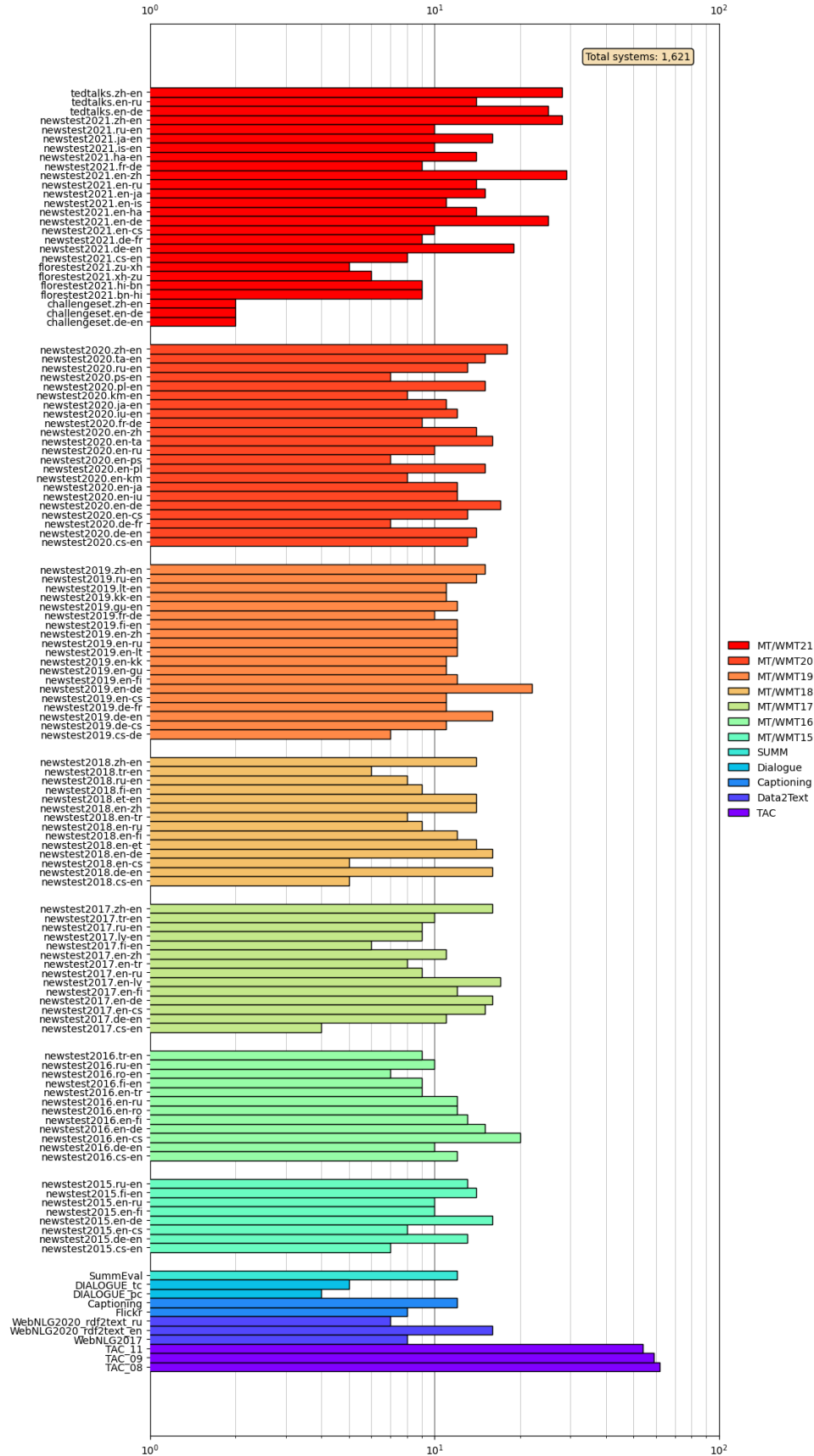


Figure 7: Number of systems in each dataset (log scale)



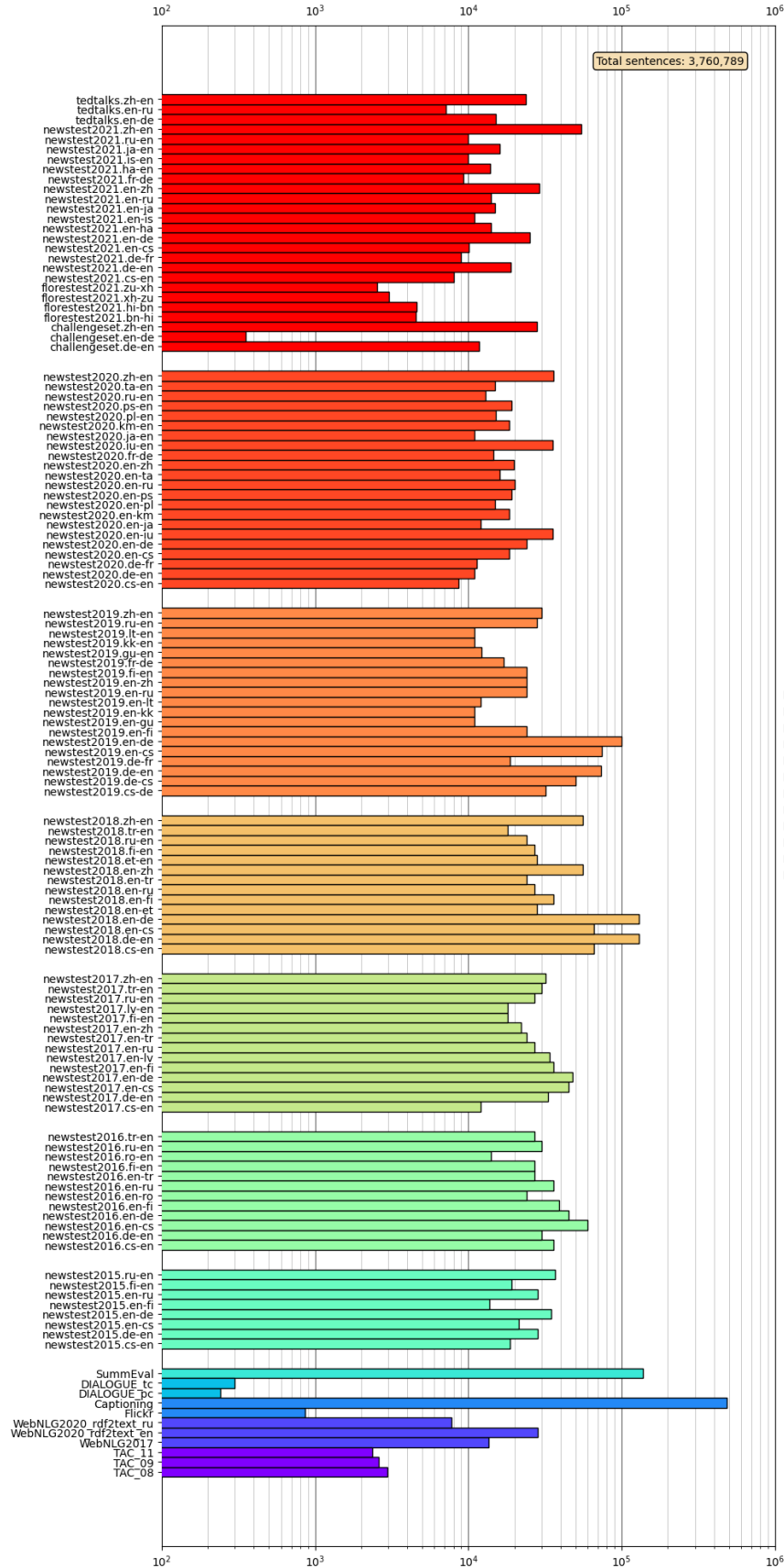


Figure 8: Number of sentences in each dataset (log scale)

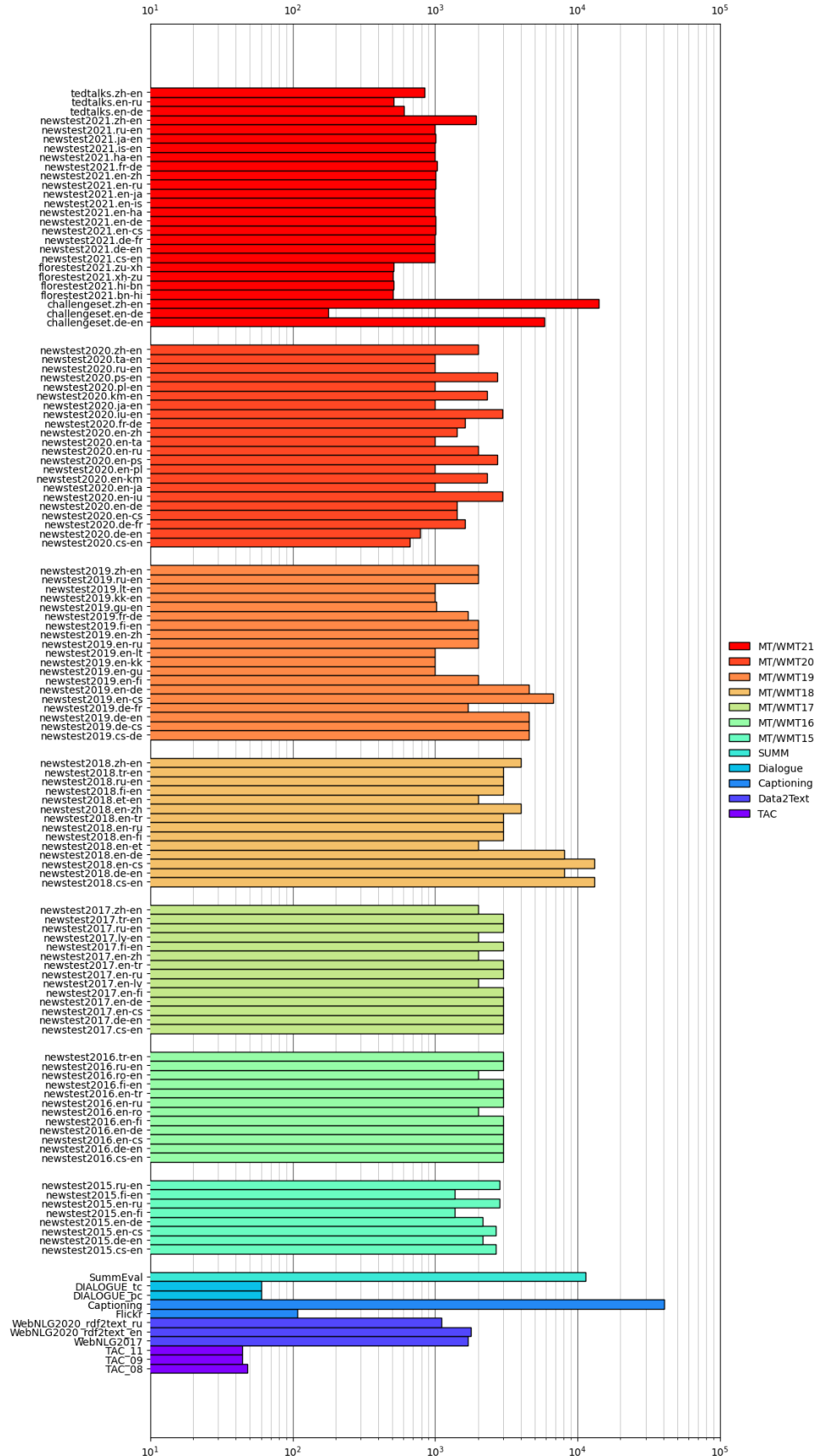


Figure 9: Average number of sentences per system in each dataset (log scale)