IASD LLM - Project Report

Camille DUBOIS, Alejandro JORBA, Killian VARESCON

- AlignScore: Evaluating Factual Consistency with a Unified Alignment Function - Yuheng Zha, Yichi Yang, Ruichen Li, Zhiting Hu - 2023 -

Contents

1 Introduction

In this report, we are reviewing (Yuheng Zha Yichi Yang, Ruichen Li and Zhiting Hu, 2023, AlignScore: Evaluating Factual Consistency with a Unified Alignment Function [2]). We first summarize the content of the article, before providing a critical analysis of the paper's content, methods and results, eventually highlighting our interest for the concepts developed in the article and our doubts regarding the final results. Finally, we compare the performance described in the paper with the results we obtained by running the same benchmarks.

2 Paper Review

2.1 Paper Presentation

Overview This paper introduces a new metric, ALIGN-SCORE [2], to evaluate the performance of LLMs (large language models). Unlike other alignment models, which are trained on task-specific datasets such as NLI or QA, ALIGN-SCORE is trained on a variety of datasets from different tasks, making it suitable for more scenarios.

The paper begins by unifying 15 datasets from 7 different language tasks, including NLI, QA, paraphrasing, fact verification, information retrieval, semantic similarity, and summarization. ALIGN-SCORE is then built using a unified alignment function, especially designed to handle long texts, and fine-tuning the RoBERTa models (125M and 355M).

It claims to achieve at least similar performance to models that are orders of magnitude larger, such as GPT-4, on state-of-the-art large-scale evaluation benchmarks.

Unified Alignment Function The model is trained on datasets from different language tasks, with varying input and output formats. Unifying these tasks is a major challenge, requiring a conceptualized unified alignment. Given two pieces of text, a and b, b is considered aligned with a if all information in b is present in a and does not contradict a. This can be interpreted as mapping text pairs (a, b) to a label y that characterizes the level of alignment:

$$f : (a, b) \mapsto y.$$

In practice, unifying the alignment of the 7 tasks mentioned above (NLI, QA, paraphrasing, fact verification, information re-

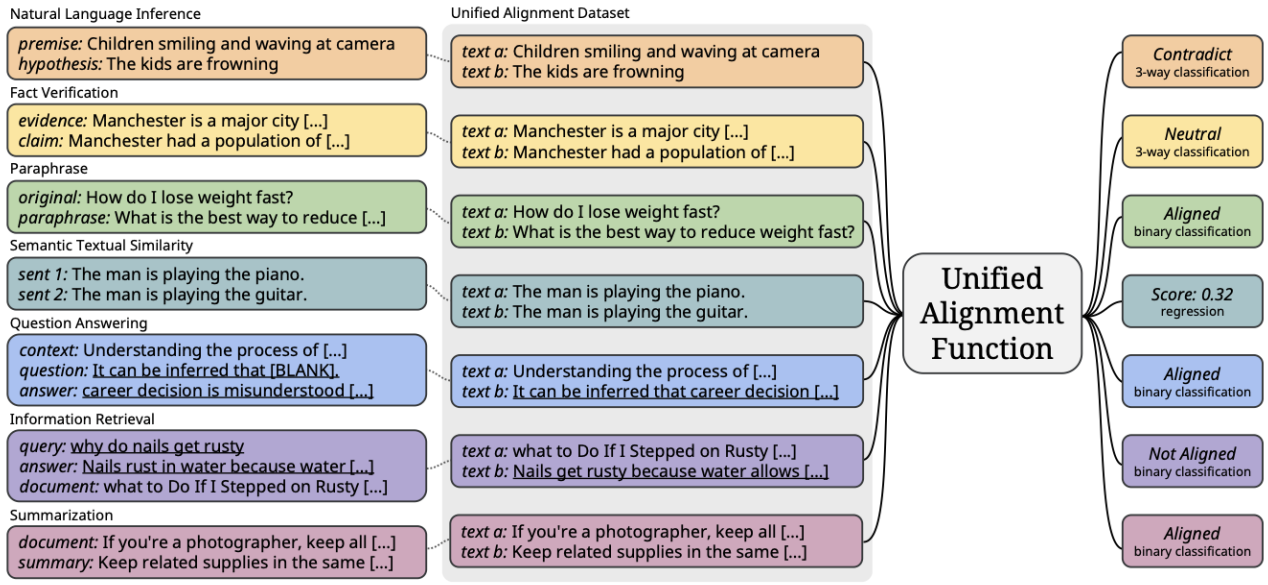

Figure 1: This schema shows how the NLP tasks are unified into the alignment task.

trieval, semantic similarity, and summarization) presents several challenges:

- QA and information retrieval do not fit the text-pair format (they contain three pieces of text, as context is added to the question-answer and query-answer pairs). Therefore, these pairs are merged into a single declarative sentence using a sequence-to-sequence model, resulting in a context-claim pair.

- Similarly, all the above-mentioned tasks do not have the same output: binary (aligned or not aligned; for tasks such as paraphrasing, QA, information retrieval, summarization), 3-way classification (aligned, contradict, or neutral; for NLI and fact verification), and regression (in $[0, 1]$; for semantic textual similarity). Three individual layers, corresponding to the three label types $(y_{bin}, y_{3-way}, y_{reg})$, are then added on top of the model, eventually leading to a joint loss function for the training part:

$$\mathcal{L}_{total} = \lambda_1 \mathcal{L}_{3-way} + \lambda_2 \mathcal{L}_{bin} + \lambda_3 \mathcal{L}_{reg}.$$

Challenges A naive approach to the problem would be to evaluate (*context, claim*) pairs. However, this approach has several disadvantages, the three biggest ones being:

- A length limitation on the input (a maximum of 512 tokens in RoBERTa), mainly affecting the context that may be truncated.

- This limitation affects the efficiency of the model, as information contained in claims can be evaluated with respect to different parts of the context (there is not necessarily a local structure).

- Classification metrics (such as those relying on 3-way labels) are not similar to the way humans perceive consistency scores—we would rather assign continuous values.

The first issue is tackled by splitting the context into chunks of upper-bounded length (typically 350 tokens) and claims into sentences, such that the concatenation of context and claim does not exceed the input-length limit. Moreover, it is said that this upper-bound is large enough to allow claims to be compared to context chunks that are

long enough to be relevant, addressing the second drawback. The third challenge is tackled by comparing each *claim* with each *context* chunk. For each *claim* sentence, the maximum over all *context* alignment scores is assigned. These are then averaged over all the *claim* sentences, yielding a more complex score that represents with better precision how the claim is supported in the *context*. It can be interpreted as the proportion of the claim that is supported in the context.

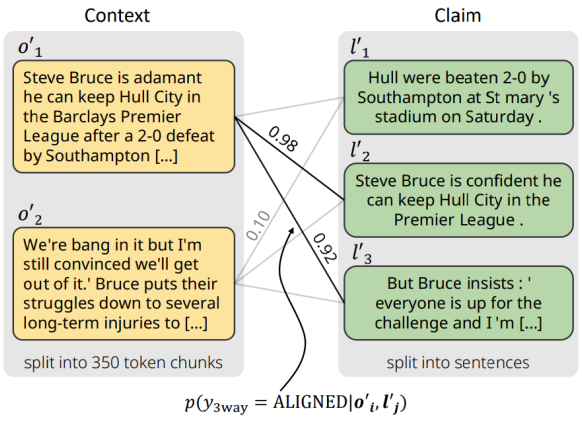

Figure 2: How ALIGN-SCORE works for a (*context, claim*) pair. Each *claim* is evaluated with respect to each *context* chunk, the highest value is then selected and averaged over all *claim* sentences to obtain the final score.

The Metric ALIGN-SCORE is defined as:

$$ALIGNSCORE(o, l) = \underset{j}{\text{mean}} \max_{i} \text{alignment}(o'_i, l'_j),$$

where (o, l) is the (*context*, *claim*) pair; and $(o'_i, l'_j)_{i,j}$ are the pairs of (*context* chunk, *claim* sentence). The alignment function is captured on the $3 - way$ classification layer of the output, representing the probability that the label $ALIGNED$ is predicted. It is chosen over binary classification and regression after experimental tests.

Training A fine-tuned RoBERTa base/large is used to implement the model. It is trained for 3 epochs with a batch size of 32 on a total of 4.7 million training samples representing diverse NLP tasks.

Results The paper claims to achieve better (or at least similar) results than other metrics on factual consistency benchmarks such as TRUE or SummaC [1]. We compared the paper's results on the latter in Section 3.

The researchers compare their ALIGN-SCORE metric with other metrics adapted from specific tasks, such as NLI, to fit with the factual consistency evaluation. AUC-ROC is used to estimate the performance of the metrics on the benchmarks.

2.2 Paper Analysis

We found the idea behind the paper interesting and original, as it carries the useful idea of evaluating LLMs' performance on a unified set of tasks. This is, in our opinion, a well-motivated problem since it can be used to assess factual consistency in a large variety of tasks, standardizing the evaluation of these performances.

The key concept of unifying different types of inputs/outputs, especially by using a three-layer output to address the compatibility problem, was an important learning for us. We also note that, as natural as it may seem, using a Large Language Model (LLM) to learn the metric was a clever choice.

However, we found the results' comparison and interpretation quite limited: as the other metrics used as baselines were task-specific, they may induce biases and be focused or built to evaluate factual consistency in a different context. Instead of comparing this metric with other task-specific metrics, for which we don't believe there is a useful application, we thought that it may have been interesting to use ALIGN-SCORE to compare one LLM's factual consistency performance on different tasks. "Does this LLM hallucinate more on paraphrasing or summarizing tasks?" could be

the kind of questions this metric could be used to answer.

While the writing was mostly clear, the GitHub repository was very poorly formatted—we had to debug it, eventually sending a pull request—which made the replicability and external evaluation of the results complex. We could, however, run our tests on the SummaC benchmark, showing coherent results with those of the paper (see Section 3 below). Nevertheless, these benchmarks do not change our opinion regarding the relevance of the baseline metrics used. To highlight our doubts, if one takes a QA task, the factual consistency will be evaluated solely with respect to the answer (e.g. FEQA); whereas in the paper, turning the (*context, question, answer*) triplet into a (*context, claim*) pair changes the perspective. Running the tests on other datasets, more convenient for the task-specific baseline metrics, could have been interesting.

Overall, as much as we found the idea and the conceptualization interesting, we are doubtful regarding the results and not convinced by the implementation methodology. We think that the addressed problem is important and that it could lead to useful practical applications, but that other iterations and implementation/evaluation methods should be explored.

3 Implementation

3.1 Setup

In our implementation of the paper, we decided to replicate the results obtained using the proposed benchmarks. We decided to focus on the SummaC benchmark, and selected a subset of the state-of-the-art factual consistency metrics that ALIGNSCORE was being compared against. Specifically, these metrics were grouped into categories: QA Based Metrics, Similarity Matching Based Metrics, Regression Based Metrics, NLI Based Metrics, and Miscellaneous. We included one metric from each of these categories in our tests: FEQA (QA), BERTScore (Similarity Matching), BLEURT (Regression), MNLI (NLI), BLANC (Miscellaneous). We also decided to focus solely on the ALIGNSCORE-large metric (based on RoBERTa-large).

Both the code for AlignScore and SummaC were publicly available (https://github.com/yuh-zha/AlignScore and https://github.com/tingofurro/summac, respectively). However, to ensure ease of execution and interpretability, we simplified the codebase, removing unnecessary complexities while preserving its core functionalities. Our implementation is available at https://github.com/alejorba/AlignScore-summac-benchmark.

3.2 Challenges

Package Compatibility Issues There were some conflicts with some key libraries employed in the AlignScore and SummaC implementations. We spent a considerable amount of time trying to solve these conflicts and "harmonize" package versions. We included all the necessary packages along with their versions in the `requirements.txt` of our repository.

Dataset Availability and Bugs The FactCC dataset originally used in the SummaC benchmark was no longer available in the official paper GitHub repository. We were not able to find any this dataset from a reputable source, so we decided to exclude it from our tests. In addition, other datasets were not available on the same sources present on the SummaC GitHub, like SummEval. We

looked for the official paper implementations of these datasets and updated our codebase accordingly.

The SummaC implementation contained a major bug that occured when instantiating the main class responsible for downloading the datasets. We solved this bug and made a pull request on the GitHub repository, see https://github.com/tingofurro/summac/pull/23

There were also some bugs related to the selected metrics. We originally intended on using FEQA from the QA category, but encountered some issues implementing it. We tried using instead the other QA metrics, but obtained similar results. Similarly, we were not able to run BLEURT from the Regression category. We decided not to spend additional time on debugging the code for these metrics and exclude them from our tests, since they were outside the scope of the original AlignScore paper implementation.

3.3 Experimental Results

The results we obtained on our benchmarks (see Table 1) show great coherence with those shown in the paper on most of the dataset (see Table 2), exception made of the PolyTope dataset, where BERTScore and ALIGNSCORE-large obtain results that are 25-40% degraded. For the SummaC dataset, the results differed at most 10% with those presented in the paper. For all the other datasets, however, the deviations were lower than or equal to 0.1%.

Type	**Metric**	**CGS**	**XSF**	**PolyTope**	**SummEval**	**FRANK**
Similarity Matching	BERTScore	63.1	49.0	61.3	70.1	84.8
NLI	MNLI	44.9	46.6	51.1	45.5	59.4
Misc	BLANC	54.1	53.5	70.4	60.5	83.4
AlignScore	ALIGNSCORE-large	86.4	75.7	53.3	81.0	91.4

Table 1: Results obtained on our implementation of the SummaC benchmark

Type	**Metric**	**CGS**	**XSF**	**PolyTope**	**SummEval**	**FRANK**
Similarity Matching	BERTScore	63.1	49.0	85.3	79.6	84.9
NLI	MNLI	44.9	46.6	45.0	43.5	59.3
Misc	BLANC	54.1	53.5	74.7	68.6	83.4
AlignScore	ALIGNSCORE-large	86.4	75.8	92.4	91.7	91.4

Table 2: SummaC benchmark results presented on the paper [extract]

3.4 Usage of AI assistants

We used AI assistants to aid us in code development, notably in identifying potential bugs in code and clarifying some doubts. We also used them sparingly to help us in the redaction of this report.

References

[1] Philippe Laban, Tobias Schnabel, Paul N. Bennett, and Marti A. Hearst. Summac: Re-visiting nli-based models for inconsistency detection in summarization. *CoRR*, abs/2111.09525, 2021.

[2] Yuheng Zha, Yichi Yang, Ruichen Li, and Zhiting Hu. Alignscore: Evaluating factual consistency with a unified alignment function, 2023.