Automated Alignment between Elicitation Interviews and Requirements

Francesco Dente
Dept. of Data Science
EURECOM
Biot, France
francesco.dente@eurecom.fr

Fabiano Dalpiaz
Dept. of Information and Computing Sciences
Utrecht University
Utrecht, The Netherlands
f.dalpiaz@uu.nl

Paolo Papotti
Dept. of Data Science
EURECOM
Biot, France
paolo.papotti@eurecom.fr

***Abstract*—Software requirements are derived from a variety of elicitation techniques, many of which have a conversational nature, like interviews. However, evaluating whether those derived requirements faithfully reflect the stakeholders' needs remains a challenging manual task. In this paper, we formalize the task of aligning the transcript of an interview with a collection of requirements represented as user stories. We propose two heuristic metrics for alignment, called (i) *requirements faithfulness*: the proportion of stories supported by the transcript, and (ii) *interview coverage*: the proportion of transcript supported by at least one story. Then, we run experiments with large language models and embedding models that assess the ability of evaluating these metrics automatically. Experiments over four datasets show that an LLM-based solution achieves 0.86 macro-F1 on manually labeled chunk–story pairs. We also show how embedding models can be used as blockers to make the approach more scalable. This work paves the way for more research on linking conversational artifacts with requirements. The formal framework and the automated matching techniques are basic components that can be used for emerging tasks such as tracing requirements to interviews and generating requirements from conversations.**

***Index Terms*—requirements interviews, user stories, alignment, traceability, large language models.**

I. Introduction

User stories are a frequently used artifact for expressing requirements in agile development [1]. These user-oriented requirements capture *who* wants *what* and *why* [2]. The most widespread notation for writing user stories is the Connextra template: "*As a [role], I want to [action], so that [benefit]*".

User stories, like other types of requirements, are commonly distilled *manually* from lengthy, conversational sources such as stakeholder interviews and facilitated workshops [3].

Recent research has leveraged Large language models (LLMs) to streamline this tedious human activity by generating candidate user stories from unstructured interview transcripts [4]–[6], or even to explore LLM's ability to act as an elicitor who conducts interviews and derives requirements [7].

Both with human or automated generation, a research question (RQ) remains largely unaddressed: *to what extent do the generated user stories faithfully reflect what stakeholders actually said?* Existing quality frameworks for user stories, like QUS [8], only assess how well a story is written, not whether it is *grounded in the source*. Addressing this question is critical, because hidden and incomplete requirements are one of the most frequent causes for software project failure [9].

We address this gap by introducing INTER2US, an NLP4RE task [10] that is concerned with measuring *interview-to-story alignment*. The goal is to quantify alignment between an interview transcript and a set of user stories derived from it. We measure alignment via two interpretable, source-faithful measures: (i) *requirements faithfulness*, the proportion of stories supported by at least one segment of the transcript, and (ii) *interview coverage*, the proportion of transcript segments that are covered by at least one story. As depicted in Fig. 1, given an interview's transcript and the set of corresponding stories, the goal is to output the two measures for an actionable quality assessment of the stories at hand.

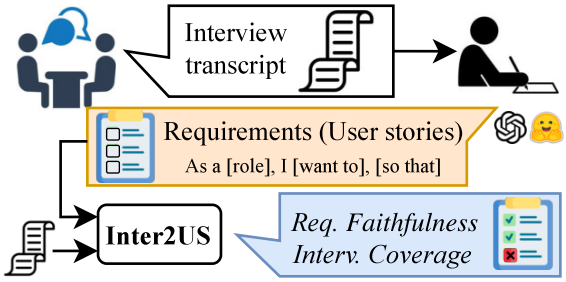

Fig. 1: Overview of the workflow. Elicitation interviews produce a transcript. From this transcript, human analysts (or LLMs) craft user stories. Given a set of stories and an interview transcript, INTER2US computes two metrics: *Requirements faithfulness* and *Interview Coverage*.

This paper makes four contributions that pave the way for research about the alignment of interview transcripts and generated requirements:

1) We formalize INTER2US as a matching problem between conversation transcript *chunks* and user stories, as depicted in Fig. 2. Segmenting interviews into overlapping chunks enables local grounding and traceability: each positive match points to concrete evidence and can be surfaced as a justification during review [11].
2) To make INTER2US practical, we introduce an *embedding-based blocking* scheme [12] that reduces the number of comparisons. Indeed, naively evaluating all

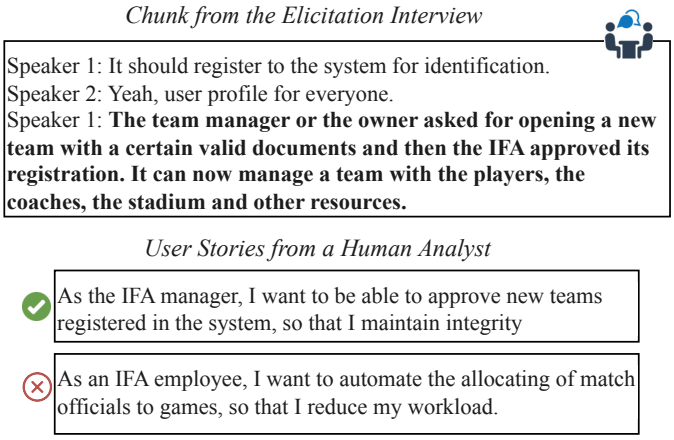

Fig. 2: Example of Interview-to-Story (INTER2US) alignment. A snippet from the elicitation interview (top) is *matched* against two candidate user stories by a *human analyst* (bottom). The former is a *valid match*, the latter is *not a match*.

story-chunk pairs scales quadratically and can become cost prohibitive for long interviews or large story sets.

3) We conduct experiments on 17 software projects and show that an LLM-based matcher achieves an average macro-F1 of 0.86 on four manually annotated datasets for the matching task.
4) Using the top-performing matcher and the non-annotated datasets, we demonstrate that *faithfulness* and *coverage* can be used to compare generated user stories (both human- and LLM-written).

After discussing related work in Sect. II, we introduce the INTER2US task in Sect. III. We present our approach for addressing INTER2US in Sect. IV, describe the experimental setup in Sect. V, and report on the results in Sect. VI. We conclude with a discussion of limitations and by sketching a research roadmap in Sect. VII.

II. RELATED WORK

We cast a common but under-formalized software engineering activity into a precise NLP4RE task, involving methods from different research areas.

Requirements engineering (RE). User stories are central to agile RE [1], [13], and their textual quality is typically assessed with frameworks such as QUS [8]. While valuable, these criteria target how well a story is written (syntax, semantics, pragmatics) rather than whether the story is *grounded* in the elicitation source. Closer to our goal, Spijkman *et al.* link user stories to elicitation conversations [11] and study how to summarize conversation transcripts to surface requirements-relevant turns [14]. INTER2US builds on this work by (i) formalizing alignment *between* stories and interview chunks and (ii) turning grounding into explicit, coverage-style metrics rather than heuristics on isolated stories.

Faithfulness and factuality. Existing work measures whether generated text is faithful to its source. In summarization, hallucinations motivate source-grounded evaluation [15]–[17]; automatic metrics rely on NLI or QA signals to score consistency [18]–[23]. INTER2US also employs the judge-against-the-source approach, but differs in unit and objective: we align *requirements* to *conversational* evidence with many-to-many relations, and we report set-level coverage measures.

Work on fact verification formalizes support/contradiction with retrieved evidence [24] and detects previously fact-checked claims with matching [25], [26]. Our setting differs as (i) claims (user stories) are longer, compositional artifacts rather than atomic statements, and (ii) evidence lives in long, multi-speaker transcripts with local context and discourse.

Retrieval, blocking, and reranking. Our *embedding-based blocking* and *pairwise matching* mirror standard information retrieval practice: bi-encoders for fast similarity search [27], cross-encoder rerankers for precise scoring [28], and techniques for reducing quadratic comparisons [12]. Unlike open-domain retrieval, we do not aim at answering a query but preserving alignment *recall* at minimal token cost, so that a stronger judge can examine a pruned set of story–chunk pairs.

LLM-as-a-judge. There is evidence of strong agreement of LLMs and human preferences in grading outputs, with growing understanding of biases and mitigation [29]. INTER2US leverages LLM-as-a-judge constrained to *pairwise* decisions at the chunk level. No prior RE work focuses on *interview-to-story* alignment task with faithfulness/coverage.

III. INTER2US: INTERVIEW-TO-STORY ALIGNMENT TASK

We define the **interview-to-story alignment (INTER2US)** task as follows: given a transcript of an elicitation interview and a set of user stories derived from it, the goal is to quantify the extent to which the stories accurately reflect the information expressed by the participants during the interview.

Formally, let T be a transcript segmented into n text chunks $C = \{c_1, c_2, \ldots, c_n\}$, and let $S = \{s_1, s_2, \ldots, s_m\}$ be a set of m user stories. The goal is to determine, for each pair (c_i, s_j), whether the chunk c_i *supports* the user story s_j, i.e., whether the requirement expressed in s_j can be justified by information contained in c_i. We define two metrics:

Requirements Faithfulness: the proportion of user stories supported by at least one chunk:

$$\text{Faithfulness} = \frac{\left|\{\, s_j \in S : \exists\, c_i \in C,\ y(c_i, s_j) = 1 \,\}\right|}{|S|}, \quad (1)$$

where $y(c_i, s_j) \in \{0, 1\}$ is the *alignment label*, with 1 if the requirement in s_j is justified by information in c_i, 0 otherwise.

Interview Coverage: the proportion of chunks in the transcript that are covered by at least one story:

$$\text{Coverage} = \frac{\left|\{\, c_i \in C : \exists\, s_j \in S,\ y(c_i, s_j) = 1 \,\}\right|}{|C|}. \quad (2)$$

Coverage is a relative metric and may not reach 1.0, since not all transcript chunks necessarily express stakeholder needs. This should not be confused with requirements completeness, which is important but not covered by our heuristics.

Since interviews vary widely in length and structure, these metrics are intended for *within-interview comparisons* (rather than across-interview metrics), such as comparing alternative story sets derived from the same transcript.

IV. Our Automated Approach to Inter2US

We define our approach to solve the Inter2US task as a two-stage pipeline: (1) segmenting the transcript into chunks, and (2) aligning the chunks with user stories via a matcher. An optional blocking stage can be inserted between these steps to reduce computational cost without altering the task definition. The requirements faithfulness and interview coverage metrics are then derived from the alignment output.

Formally, let $C = \{c_1, \ldots, c_n\}$ be the set of transcript chunks and $S = \{s_1, \ldots, s_m\}$ the set of user stories. We consider the set of candidate pairs

$$P = C \times S = \{(c_i, s_j) \mid c_i \in C, s_j \in S\}.$$

A matcher X is a function

$$X : C \times S \to \{0, 1\},$$

where $X(c_i, s_j) = 1$ indicates that chunk c_i *supports* story s_j. By default, X is applied to the entire Cartesian product P, i.e., all possible pairs.

The approach admits two operating modes:

1) **Direct matching**: evaluate X on all pairs $P = C \times S$.
2) **Blocked matching**: apply an embedding-based blocking operator B_K that restricts P to a smaller candidate set P', then evaluate X only on P'.

We now detail the three core components: *chunking* (C), *pairwise matching* (X), and *embedding-based blocking* (B_K).

A. Chunking (C)

We segment transcripts based on speaker turns (as in Fig. 2), following prior work on identifying requirements information in interviews [14]. While one could in principle treat each turn as a separate chunk, elicitation interviews are generally a question-and-answer sequence [30], which provides contextual information (spanning across multiple turns) that helps understand whether a user story is supported.

Following the approach by previous RE research [14], we adopt the strategy of grouping three consecutive speaker turns into a unit, using a stride of one so that every possible three-turn window is considered and full interview coverage is ensured. Given turns $(t_1, t_2, \ldots, t_k)$, this results in overlapping chunks $(t_1, t_2, t_3), (t_2, t_3, t_4), \ldots, (t_{k-2}, t_{k-1}, t_k)$.

B. Pairwise Matching (X)

Given the chunk set C and the user stories S, we form all possible pairs (c_i, s_j). For each pair, a model X assigns a binary label indicating whether the chunk supports the story.

The matcher X can be instantiated as:

1) **LLM-based judge**, a large language model prompted with few shot examples of aligned and non-aligned pairs (see a prompt excerpt in Fig. 3 and the full prompt in the online appendix: https://zenodo.org/records/20596692);
2) **Cross-encoder**, a reranker model that computes the similarity between c_i and s_j by jointly encoding (c_i, s_j) and by returning a support score; or
3) **Bi-encoder**, which computes embeddings for c_i and s_j independently and calculates their cosine similarity.

In all cases, X answers the question: *"Does this interview chunk justify this user story?"*

System Prompt

You are a senior requirements analyst.
Your task: decide whether the **Chunk Text** gives *enough evidence* to *justify* the **User Story**.

Decision rules
– Output `1` if a knowledgeable reader could infer the User Story from the Chunk Text alone.
– Otherwise output `0`.

Note: Each Chunk Text is an excerpt from an informal interview transcript. Expect colloquial language, filler words ("uh", "you know"), and broken grammar; ignore these and focus on meaning.

Additional guidance
– User stories follow the format: `"As a <type of user>, I want to <goal> so that <reason>."`
– Pay special attention to the **goal** in the User Story.
– Verify that the **type of user** (e.g., "employee") is consistent with the Chunk Text.

Example (1 of 4)

```
User Story: As a match official, I want
to report on match events live, so that I
do not register them twice.
Chunk Text: <excerpt about referee
recording events in real time>
Answer: 1
... (three more few-shot examples omitted
for brevity) ...
```

Now answer for the next pair.
Return **only one character**, either `1` or `0`, followed by nothing else.

Fig. 3: System prompt for the LLM-based judge matcher.

C. Embedding-Based Blocking (B_K)

To reduce computational cost, we introduce an embedding-based blocking step. Instead of evaluating all pairs, we first compute embeddings for all chunks and stories and measure their similarity. For each story s_j, we keep only the Top-K most similar chunks: $B_K(S, C) = \bigcup_{j=1}^{m} \{(c_i, s_j) : c_i \in \text{TopK}(C, s_j)\}$. The matcher X is then applied only to these reduced candidate sets $P' = B_K(S, C)$. All other pairs are labeled as $X(c_i, s_j) = 0$. This blocking scheme preserves alignment quality while drastically reducing computation by a factor $\frac{|C|}{K}$. For instance, with $|C| = 100$ and $K = 25$, blocking cuts the number of model calls by $4\times$.

TABLE I: Statistics of the 17 datasets. Private datasets consist of student projects (each with two interviews).

Dataset	#Stories	#Chunks	$\lvert C \times S \rvert$ tokens	Domain
Student Project 1	53	386	3.76M	Predictive Dental Appointment Scheduler
Student Project 2	44	408	3.26M	Centralized communication system for supermarkets
Student Project 3	59	546	3.87M	Information system for modernization of bakery
Student Project 4	80	169	2.61M	Learning support information system
Student Project 5	50	564	3.78M	Bike business improvement to kickstart efficiency
Student Project 6	62	380	4.73M	Workflow management system for solar panel company
Student Project 7	82	450	5.84M	Movie and series availability system
Student Project 8	46	277	2.52M	User-friendly app for devs to upload apps on Steam
Student Project 9	57	538	4.48M	Unified digital workspace for UX agency
Student Project 10	55	201	2.57M	Information system for automated train operation
Student Project 11	71	406	4.92M	Driving schools comparator
Student Project 12	52	714	4.36M	Cinema organization
Student Project 13	121	386	9.29M	Football league operations department
Student Project 14	64	440	4.49M	Scheduling tool for pizzeria
Student Project 15	39	429	2.39M	Venue management system events coordination
Public Interview	37	132	0.88M	International football association portal
System Description	115	37	0.19M	Travel agency management system

V. Experimental Setup

Datasets. We evaluate our approach on 17 datasets (Table I). The primary source consists of 15 software projects from a requirements engineering course. In each project, students conducted elicitation interviews while role-playing stakeholders and analysts to gather requirements for a target software system, following the pedagogical approach proposed by Ferrari *et al.* [31]. For each project, we collected (i) transcripts of two elicitation interviews, concatenated into a single document, and (ii) the corresponding set of user stories. All data are in English and each project targets a distinct application domain. Students could refine requirements outside the elicitation sessions. Due to privacy constraints, these datasets cannot be released publicly and only local models can be used on them.

In addition, we include two public datasets:

1) *Public Interview Dataset*: a public interview transcript [14], combined with a set of user stories [32] for the same system, forming a transcript–stories pair comparable to our private projects.
2) *System Description Dataset*: a textual system description with corresponding user stories [33]. Note: While this dataset does not originate from interviews, we want to assess whether our method would also work for non-conversational data. Here, we cannot build chunks based on speaker turns, but instead split at newlines.

To the best of our knowledge, no public dataset explicitly annotates which segments of a natural language source (interview transcripts or descriptions) support individual user stories. Existing work either evaluates user stories in isolation [5], [6], or compares generated stories against a reference set, without grounding them in the originating text [4], [34]. Korn *et al.* [7] mark whether high-level requirements are present in a transcript, but do not identify their exact location. In contrast, our annotated datasets explicitly target *source grounding* by providing manually created alignments between user stories and corresponding chunks of the source text.

Annotation Protocol. We manually annotated matching pairs for two private student projects (from different domains) and the public datasets (Table II). Exhaustively annotating all chunk-story pairs is infeasible due to the $|C| \times |S|$ complexity, so we adopted a similarity-driven protocol:

1) For each user story, we retrieved source chunks ranked by decreasing embedding similarity using all-mpnet-base-v2;
2) We selected the top-5 non-overlapping chunks (given stride-based chunking);
3) If fewer than two positive matchings were found among the top-5, we extended the ranking until at least two positives were identified;
4) When we knew from prior reading that a specific chunk was the most suitable support, we included it even if it was not in the top candidates.

This process led to a balanced yet feasible set of annotated examples, which are used exclusively to evaluate the pairwise chunk–story matching step.

TABLE II: Statistics of the annotated test datasets used for evaluation. Each test set is a subset of the full dataset, selected and balanced via our annotation protocol.

Dataset	#Total Pairs	#Positive pairs	#Negative pairs
Student Proj. 1 (S1)	287	109	178
Student Proj. 2 (S2)	236	88	148
Public Int. (PI)	180	74	106
System Desc. (SD)	580	302	278

We assessed inter-annotator reliability on a sample (50 chunk-story couples taken from two projects) using Fleiss' κ with three annotators. The overall agreement is κ=0.470 (*moderate* agreement), indicating that agreement between humans is also challenging and opens up future directions (see Sect. VII). Additional details are in the online appendix.

Metrics. We use the following criteria:

1) *Macro F1-score*: evaluates the alignment quality of the matcher X on the annotated datasets.
2) *Faithfulness and Coverage metrics*: assessed in different scenarios to demonstrate the usefulness of our approach.
3) *Computational cost*: measured as the total number of tokens processed (see $|C \times S|$ tokens in Table I), allowing us to compare blocking and non-blocking approaches.

Implementation. Our Python code is available at https://zenodo.org/records/20596692, also acts as online appendix, as it allows reproducing the experiments with the two public datasets.

VI. Results

We first evaluate the quality of matchers for the pairwise matching task against human-annotated pairs (Sect. VI-A). Then, we show that the proposed faithfulness and coverage metrics, computed over the full set of projects with the best matcher, effectively compare different sets of user stories, including both human- and LLM-generated ones (Sect. VI-B). Next, we assess the efficiency of the embedding-based blocking strategy for the matching task (Sect. VI-C). Finally, we compare the chunking approach versus giving the entire interview transcript to the matcher (Sect. VI-D).

A. Pairwise Alignment Quality

We evaluate the quality of the pairwise matcher X (Sect. IV-B) on the manually annotated datasets, measuring macro F1-score across positive and negative pairs. Table III reports the results for the three matcher families described in Sect. IV-B: LLM judges, cross-encoders, and bi-encoders. All experiments are run with LLMs' temperature set to 0.

LLM judges emerge as the best-performing matchers. Qwen3-32B and Llama3.3-70B achieve macro-F1 scores above 0.80 across all datasets, with top average scores of 0.841 and 0.859, respectively. Cross-encoders deliver competitive results and are particularly effective at smaller model sizes; however, they output continuous scores and therefore require calibration to determine the optimal threshold that discriminates positives from negatives. For threshold-based approaches (bi-encoder and cross-encoder), Table III shows results for the best threshold per dataset, and should be interpreted as optimistic upper bounds. Finally, bi-encoder matchers perform worse overall, with low macro-F1 scores.

B. Stories' Faithfulness and Interview Coverage

We now assess the behavior of our *Faithfulness* and *Coverage* metrics to validate them as indicators of source alignment. For each of the 15 private student projects, we let Qwen models of increasing size (0.6B to 32B) generate up to 50 user stories from the interview transcripts - the cap prevents excessively long generations and mitigates hallucinated "never-ending" outputs (see the online appendix for the generation prompt). We then apply our INTER2US pipeline in direct matching mode (no B_K), using Qwen3-32B as matcher X, and compute faithfulness and coverage. Fig. 4 and Fig. 5 report the average scores across the 15 datasets, with standard deviation as error bars. We also report the metrics for the human-written stories produced by the students (*Human*). For coverage, we also report the average ranking (lower is better) across datasets.

Coverage increases with generator size under a fixed judge and pipeline. This trend is expected: larger LLMs are known to generate more diverse and exhaustive outputs and should therefore cover a larger fraction of the interview content [35]. The monotonic increase across model sizes suggests that the metric is sensitive to this external signal, a pattern that is also reflected by the average coverage ranking across datasets reported in Figure 5. LLM-generated story sets also achieve higher coverage than the human ones, as they tend to include more outputs, which humans may have overlooked.

In contrast, **faithfulness is consistently high** across all but the smallest generator (Qwen3-0.6B). This asymmetry mirrors real-world software engineering practice: elicited requirements are often individually valid, but achieving com-

TABLE III: Macro F1-scores of different matcher instantiations X on the four annotated datasets. Results for threshold-based models are reported with best threshold and marked with [t]. Best score is in bold and second best is highlighted in grey.

Matcher (X)	**Student 1**	**Student 2**	**Public Interview**	**System Description**	**Average**
Bi-encoder [t] (Qwen3-0.6B)	0.424	0.593	0.649	0.718	0.596
Bi-encoder [t] (Qwen3-4B)	0.362	0.576	0.741	0.741	0.605
Bi-encoder [t] (Qwen3-8B)	0.557	0.635	0.771	0.707	0.668
Cross-encoder [t] (Qwen3-Reranker-0.6B)	0.647	0.679	0.748	0.841	0.729
Cross-encoder [t] (Qwen3-Reranker-4B)	0.765	0.728	0.797	**0.914**	0.801
Cross-encoder [t] (Qwen3-Reranker-8B)	0.776	0.737	0.829	0.895	0.809
LLM (Qwen3-0.6B)	0.310	0.365	0.291	0.507	0.406
LLM (Qwen3-1.7B)	0.326	0.597	0.483	0.693	0.525
LLM (Qwen3-4B)	0.774	0.676	0.732	0.856	0.760
LLM (Qwen3-8B)	0.774	0.739	0.754	0.902	0.792
LLM (Qwen3-14B)	0.739	0.792	0.814	0.888	0.808
LLM (Qwen3-32B)	0.803	0.831	0.815	**0.914**	0.841
LLM (Llama3.2-1B)	0.455	0.418	0.493	0.324	0.423
LLM (Llama3.2-3B)	0.533	0.671	0.601	0.752	0.639
LLM (Llama3.1-8B)	0.578	0.696	0.703	0.863	0.711
LLM (Llama3.3-70B)	**0.818**	**0.840**	**0.876**	0.902	**0.859**

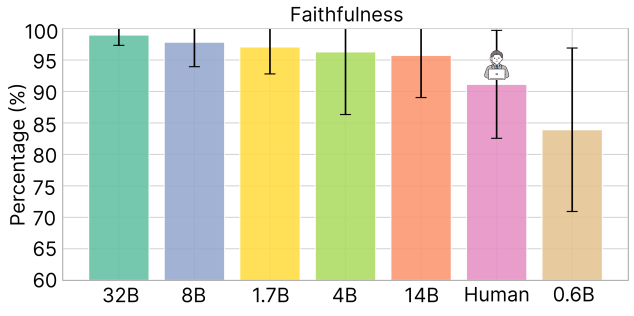

Fig. 4: Average *faithfulness* of user stories generated by Qwen models of increasing size. Error bars denote ± 1 standard deviation across datasets (n=15).

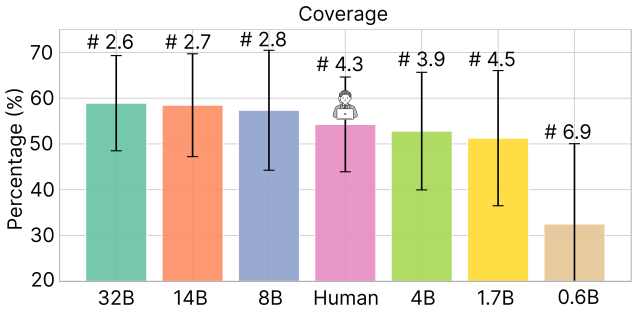

Fig. 5: Average *coverage* and model ranking (#) of user stories generated by Qwen models of increasing size. Error bars denote ± 1 standard deviation across datasets (n=15).

plete requirements (here, approximated by coverage) is a recurring challenge due to the implicit and *tacit* nature of requirements [36], [37]. Interestingly, human-written stories obtain lower faithfulness than most LLM-generated sets. This is expected in our setting, as students were allowed to rely on sources beyond the interview transcripts, leading to stories not grounded in the interview data.

C. Blocking Efficiency

We evaluate the extent to which the embedding-based blocking operator (B_K) reduces computational cost. We aim to obtain the same pairwise matching performance that would be achieved by applying a matcher to the full $|C| \times |S|$ Cartesian product, while processing fewer tokens. Blocking must retain the pairs classified as positive by the matcher on the full cartesian product; if all positive pairs are preserved, downstream faithfulness and coverage scores remain unchanged. We use Qwen3-32B as the matcher to compute the set of positive pairs from the full Cartesian product and we compare off-the-shelf Qwen3-Embedding-0.6B and its fine-tuned version. To fine-tune the embedding model, we generate training data by labeling all chunk-story pairs with the matcher and balancing positives and negatives via random downsampling. Evaluation follows a leave-one-out protocol across the 15 private projects.

Fig. 6 reports, for increasing recall levels, the fraction of tokens that must be processed after blocking. For each story,

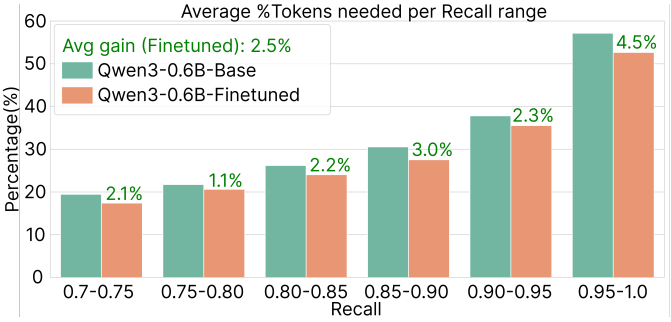

Fig. 6: Recall of positive pairs vs. percentage of tokens retrieved by the blocking operator (B_K) across datasets. Lower values indicate higher efficiency.

we retrieve the top-K most similar chunks and increase K until the desired recall with respect to the reference positives is reached. Because chunk lengths vary, we report token counts rather than K, expressed as a percentage of the Cartesian product. Blocking is efficient: on average, only one third of the tokens ($3\times$ reduction) are needed for recall 0.9–0.95, and about half ($2\times$ reduction) for recall 0.95–1. Fine-tuning further improves efficiency although not substantially.

D. Chunking vs. Feeding the Entire Interview

Chunking is key to our alignment task, where the matching is between a fragment and a user story. To evaluate its impact, we compare this pairwise approach with an ablation setting in which the model receives the *entire interview transcript*, segmented into numbered chunks (full-context). The model is asked, given a user story, to predict the indices of the supporting chunks. Transcripts are split into batches of 200 chunks to remain within context window limits.

TABLE IV: Macro F1-scores comparing full-context evaluation with the pairwise matcher on the annotated datasets (S1, S2, PI, SD). Qwen3-32B is used as matcher X.

Dataset	S1	S2	PI	SD	Avg
Full Context	0.471	0.540	0.623	0.322	0.489
Pairwise	**0.803**	**0.831**	**0.815**	**0.914**	**0.841**

As shown in Table IV, providing the model with full-context chunking degrades significantly the macro F1-score performance compared to the pairwise approach, thereby confirming the importance of chunking for the task at hand (given the limitations of existing language models).

VII. Discussion and Roadmap

We introduced Inter2US, an evaluation framework that formalizes *interview-to-story alignment* as a many-to-many matching between interview chunks and user stories. We build a pipeline that includes turn-aware chunking, a calibration-free pairwise judge, and an embedding-based blocking operator that preserves alignment recall while reducing token cost.

On four annotated sets, LLM judges attain strong alignment quality enabling the computation of two quality measures:

Requirements Faithfulness and Interview Coverage. Applying these metrics to generated stories shows that they effectively assess both human- and LLM-written user stories.

A. Threats to Validity

We discuss the main factors that may affect the results reported in this paper. Additional limitations are treated as opportunities and discussed in the roadmap.

Our *interview coverage* is not equivalent to requirements completeness, which would be the ultimate metric to calculate in addition to faithfulness. A rigorous correlation study aligning transcript coverage with human expert-validated requirements would be valuable to establish to what extent our metric correlates with completeness.

Our metrics are not a universal measure of user story quality. INTER2US focuses on the *source alignment* aspect and should be seen as a complement to existing quality frameworks such as QUS [8]. While QUS criteria capture dimensions such as atomicity, estimability, uniqueness, and other syntactic, semantic and pragmatic criteria, our metric does not evaluate how well a story is written or whether it follows the user story template. Currently, comparison with other metrics is difficult because there is no baseline for measuring how well a set of user stories reflects the content of stakeholder interviews.

Our evaluation datasets are limited in scope. We rely primarily on student projects, which may not fully reflect industrial elicitation practices. Only two datasets are publicly available, both annotated in terms of matching by one of the authors and cross-checked with the other two on a subset, which may introduce annotation noise. Some experiments cannot be fully reproduced because certain transcripts cannot be released due to confidentiality constraints. To keep manual annotation feasible, we used embeddings to limit the number of chunks to annotate; this may have led to an imperfect ground truth.

This paper does not aim at evaluating the effectiveness of prompt engineering, nor the suitability of certain prompts to specific models. We created simple prompts and we made them available. Nevertheless, we acknowledge that model performance could be affected by the selected prompts [38].

B. Roadmap

The main contribution of this paper is that of formalizing a new NLP task for RE, besides classic ones [10] such as ambiguity detection, model generation, and trace link recovery. The reported results are only a starting point and our research unveils numerous directions at the intersection of elicitation conversations and documented requirements.

Aligning with other requirements and information types. We only focused on user-oriented requirements, particularly through the notation of user stories. Although this kind of requirements is prevalent in agile development artifacts like backlog items [39], there are other types of requirements, including non-functional ones, which could be traced back to their source. Furthermore, as shown by Spijkman *et al.* [40], requirements interviews may include additional information (like descriptions of as-is processes) that goes beyond requirements. Therefore, studying the matching of these additional requirements and information types would lead to a more comprehensive overview of alignment.

Requirements generation via LLMs. We only touched the surface of a hot topic [5], [7]: the LLM-based generation of requirements. We did so in the context of evaluating the applicability of our metrics. The use of LLMs, and generative AI at large, for the goal of generating requirements is a very important direction, guided by questions concerning 'How well do LLMs perform compared to humans?' To answer this type of questions, we need studies that focus on experimenting with a number of LLMs as well as prompting patterns, and that do not only take a quantitative standpoint, but that also account for qualitative differences (error profiles). Moreover, since requirements are only the inception of software development, it would be interesting to study how LLM-generated requirements affect downstream activities.

Revised Quality Frameworks. While studying techniques for generating requirements from conversations, we expect revived interest in quality frameworks. We surmise that classical qualities of requirements, including adherence with a template, atomicity, or non-vagueness, can easily be achieved via language model via adequate prompting. Therefore, rather than focusing on these quality criteria, we would expect frameworks and automated tools that are able to focus on more advanced quality criteria that have to do with semantics, such as conflicts or difficult-to-estimate requirements.

What ground truth? While advances in AI research are also supported by the establishment of ground-truth benchmarks, these are hardly existent when it comes to elicitation conversations and their requirements. First, very few requirements interviews are publicly available, and a subset of those have a corresponding specification. Second, there is no single set of requirements for a given conversation, and this makes alignment a complex challenge. As such, future research should explore in depth this issue, e.g., via user studies where multiple analysts and LLMs are required to generate requirements from the same interview.

Validation in industry. We primarily analyzed conversations and requirements generated by groups of students in university projects. In addition to not being representative of the ability of a seasoned professional, all these interviews were conducted in a uniform setting: two interviews of roughly one hour each, focus on an information system, students instructed by the same lecturer, etc. We expect that validation in industry will reveal a high variety of interview types and of specification formats. This would call for approaches and evaluations that do not only consider the ability of generating *one* type of requirements starting from *one* kind of interview, but also their generality and adaptability.

ACKNOWLEDGEMENT

This work was partially supported by the AI4SWEng project, funded by the EU's Horizon program (grant agreement No. 101189909), and by the 3IA Côte d'Azur Investments in the IA-cluster project managed by the National Research Agency (ANR) with the reference number ANR-23-IACL-0001. We would like to thank the students who voluntarily provided their project data, thereby allowing our experiments.

REFERENCES

[1] G. Lucassen, F. Dalpiaz, J. M. E. v. d. Werf, and S. Brinkkemper, "The use and effectiveness of user stories in practice," in *Proc. of REFSQ*, ser. LNCS, vol. 9619. Springer, 2016, pp. 205–222.

[2] M. Cohn, *User Stories Applied: For Agile Software Development*. Addison-Wesley Professional, 2004.

[3] S. Wagner, D. M. Fernández, M. Felderer, A. Vetrò, M. Kalinowski, R. Wieringa, D. Pfahl, T. Conte, M.-T. Christiansson, D. Greer, T. M. Casper Lassenius, M. Nayebi, M. Oivo, B. Penzenstadler, R. Prikladnicki, G. Ruhe, A. Schekelmann, S. Sen, R. Spínola, A. Tuzcu, J. L. de La Vara, and D. Winkler, "Status quo in requirements engineering: A theory and a global family of surveys," *ACM Transactions on Software Engineering and Methodology*, vol. 28, no. 2, pp. 1–48, 2019.

[4] K. Ronanki, B. Cabrero-Daniel, and C. Berger, "ChatGPT as a tool for user story quality evaluation: Trustworthy out of the box?" in *Proc. of XP*, ser. LNBIP, vol. 489. Springer, 2022, pp. 173–181.

[5] G. Quattrocchi, L. Pasquale, P. Spoletini, and L. Baresi, "Can LLMs generate user stories and assess their quality?" *arXiv preprint arXiv:2507.15157*, 2025.

[6] A. Yamani, M. Baslyman, and M. Ahmed, "Leveraging LLMs for user stories in AI systems: UStAI dataset," in *Proc. of PROMISE*. ACM, 2025, pp. 21–30.

[7] A. Korn, S. Gorsch, and A. Vogelsang, "LLMREI: Automating requirements elicitation interviews with LLMs," in *Proceedings of RE*. IEEE, 2025, pp. 19–30.

[8] G. Lucassen, F. Dalpiaz, J. M. E. M. van der Werf, and S. Brinkkemper, "Improving agile requirements: The Quality User Story framework and tool," *Requirements Engineering*, vol. 21, pp. 399–417, 2016.

[9] D. M. Fernández, S. Wagner, M. Kalinowski, M. Felderer, P. Mafra, A. Vetrò, T. Conte, M.-T. Christiansson, D. Greer, C. Lassenius, T. Männistö, M. Nayabi, M. Oivo, B. Penzenstadler, D. Pfahl, G. R. Rafael Prikladnicki, A. Schekelmann, S. Sen, R. Spinola, A. Tuzcu, J. L. de La Vara, and R. Wieringa, "Naming the pain in requirements engineering: Contemporary problems, causes, and effects in practice," *Empirical Software Engineering*, vol. 22, no. 5, pp. 2298–2338, 2017.

[10] L. Zhao, W. Alhoshan, A. Ferrari, K. J. Letsholo, M. A. Ajagbe, E.-V. Chioasca, and R. T. Batista-Navarro, "Natural language processing for requirements engineering: A systematic mapping study," *ACM Computing Surveys (CSUR)*, vol. 54, no. 3, pp. 1–41, 2021.

[11] T. Spijkman, F. Dalpiaz, and S. Brinkkemper, "Back to the roots: Linking user stories to requirements elicitation conversations," in *Proc. of RE*. IEEE, 2022, pp. 281–287.

[12] G. Papadakis, D. Skoutas, E. Thanos, and T. Palpanas, "Blocking and filtering techniques for entity resolution: A survey," *ACM Computing Surveys*, vol. 53, no. 2, Mar. 2020.

[13] M. Kassab, "An Empirical Study on the Requirements Engineering Practices for Agile Software Development," in *Proc. of EUROMICRO-SEAA*, 2014, pp. 254–261. [Online]. Available: http://ieeexplore.ieee.org/document/6928819/

[14] T. Spijkman, X. de Bondt, F. Dalpiaz, and S. Brinkkemper, "Summarization of elicitation conversations to locate requirements-relevant information," in *Proc. of REFSQ*, ser. LNCS, vol. 13975. Springer, 2023, pp. 122–139.

[15] J. Maynez, S. Narayan, B. Bohnet, and R. McDonald, "On faithfulness and factuality in abstractive summarization," in *Proc. of ACL*. ACL, 2020, pp. 1906–1919.

[16] A. Pagnoni, V. Balachandran, and Y. Tsvetkov, "Understanding factuality in abstractive summarization with FRANK: A benchmark for factuality metrics," in *Proc. of NAACL*. ACL, 2021, pp. 4812–4829.

[17] A. Nenkova and R. Passonneau, "Evaluating content selection in summarization: The pyramid method," in *Proc. of HLT-NAACL*. Association for Computational Linguistics, 2004, pp. 145–152.

[18] W. Kryściński, B. McCann, C. Xiong, and R. Socher, "Evaluating the factual consistency of abstractive text summarization," in *Proc. of EMNLP*. ACL, 2020, pp. 9332–9346.

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

[20] A. Wang, K. Cho, and M. Lewis, "Asking and answering questions to evaluate the factual consistency of summaries," in *Proc. of ACL*. ACL, 2020, pp. 5008–5020.

[21] T. Scialom, P.-A. Dray, S. Lamprier, B. Piwowarski, J. Staiano, P. Gallinari, and I. Sorodoc, "QuestEval: Summarization asks for fact-based evaluation," in *Proc. of EMNLP*. ACL, 2021, pp. 1312–1323.

[22] A. Fabbri, C.-S. Wu, W. Liu, and C. Xiong, "QAFactEval: Improved qa-based factual consistency evaluation for summarization," in *Proc. of NAACL*. ACL, 2022, pp. 2587–2601.

[23] Y. Zha, Y. Yang, R. Li, and Z. Hu, "AlignScore: Evaluating factual consistency with a unified alignment function," in *Proc. of ACL*. Association for Computational Linguistics, 2023.

[24] P. Nakov, D. P. A. Corney, M. Hasanain, F. Alam, T. Elsayed, A. Barrón-Cedeño, P. Papotti, S. Shaar, and G. D. S. Martino, "Automated fact-checking for assisting human fact-checkers," in *Proc. of IJCAI*, 2021, pp. 4551–4558.

[25] M. Saeed, N. Traub, M. Nicolas, G. Demartini, and P. Papotti, "Crowdsourced fact-checking at twitter: How does the crowd compare with experts?" in *Proc. of CIKM*. ACM, 2022, pp. 1736–1746.

[26] S. Shaar, N. Babulkov, G. D. S. Martino, and P. Nakov, "That is a known lie: Detecting previously fact-checked claims," in *Proc. of ACL*, 2020, pp. 3607–3618.

[27] N. Reimers and I. Gurevych, "Sentence-BERT: Sentence embeddings using Siamese BERT-networks," in *Proc. of EMNLP-IJCNLP*. ACL, 2019, pp. 3982–3992.

[28] R. Nogueira and K. Cho, "Passage re-ranking with BERT," *arXiv preprint arXiv:1901.04085*, 2019. [Online]. Available: https://arxiv.org/abs/1901.04085

[29] L. Zheng, W.-L. Chiang, Y. Sheng, S. Zhuang, Z. Wu, Y. Zhuang, Z. Lin, Z. Li, D. Li, E. P. Xing, H. Zhang, J. E. Gonzalez, and I. Stoica, "Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena," 2023, NeurIPS 2023 Datasets & Benchmarks track.

[30] O. Zaremba and S. Liaskos, "Towards a typology of questions for requirements elicitation interviews," in *Proc. of RE*. IEEE, 2021, pp. 384–389.

[31] A. Ferrari, P. Spoletini, M. Bano, and D. Zowghi, "Sapeer and reverse-esapeer: teaching requirements elicitation interviews with role-playing and role reversal," *Requirements Engineering*, vol. 25, no. 4, pp. 417–438, 2020.

[32] F. Dalpiaz, A. Sturm, and P. Gieske, "Extraction of conceptual models: User stories vs. use cases," Zenodo dataset, 2020. [Online]. Available: https://doi.org/10.5281/zenodo.4121935

[33] R. Santos, G. Freitas, I. Steinmacher, T. Conte, A. C. Oran, and B. Gadelha, "User stories: Does ChatGPT do it better?" in *Proc. of ICEIS*. SciTePress, 2025, pp. 47–58.

[34] A. Sharma and A. Kumar Tripathi, "Evaluating user story quality with LLMs: A comparative study," *Journal of Intelligent Information Systems*, vol. 63, pp. 1423–1451, 2025.

[35] J. Kaplan, S. McCandlish, T. Henighan, T. B. Brown, B. Chess, R. Child, S. Gray, A. Radford, J. Wu, and D. Amodei, "Scaling laws for neural language models," *arXiv preprint arXiv:2001.08361*, 2020.

[36] A. Sutcliffe and P. Sawyer, "Requirements elicitation: Towards the unknown unknowns," in *Proc. of RE*. IEEE, 2013, pp. 92–104.

[37] A. Ferrari, P. Spoletini, and S. Gnesi, "Ambiguity and tacit knowledge in requirements elicitation interviews," *Requirements Engineering*, vol. 21, no. 3, pp. 333–355, 2016.

[38] K. Ronanki, B. Cabrero-Daniel, J. Horkoff, and C. Berger, "Requirements engineering using Generative AI: Prompts and prompting patterns," in *Generative AI for Effective Software Development*. Springer, 2024, pp. 109–127.

[39] A. T. van Can and F. Dalpiaz, "Locating requirements in backlog items: Content analysis and experiments with large language models," *Information and Software Technology*, vol. 179, p. 107644, 2025.

[40] T. Spijkman, F. Dalpiaz, and S. Brinkkemper, "Requirements elicitation via fit-gap analysis: A view through the grounded theory lens," in *International Conference on Advanced Information Systems Engineering*. Springer, 2021, pp. 363–380.