Orchestrator RegComAgent: An Agentic AI Framework for Multilingual ESG Regulatory Compliance Verification

Wei Chen Huang[1], Hsin Ting Lu[2], and Min Yuh Day[3]*

[1] Department of Statistics, National Taipei University, New Taipei City, Taiwan
wesleyseawatch@gmail.com
[2] Graduate Institute of Information Management, National Taipei University, New Taipei City, Taiwan
hsintinglubob@gmail.com
[3] Graduate Institute of Information Management, National Taipei University, New Taipei City, Taiwan
myday@gm.ntpu.edu.tw

Abstract. Sustainability disclosure regulations increasingly require firms to report ESG information according to standardized metric definitions, disclosure categories, and units of measure, yet ESG reports are often long, multilingual, and semi-structured, making manual page-level compliance review costly and inconsistent. This study addresses Subtask 2 of the NTCIR 19 RegCom shared task, where a system receives one SASB metric and one ESG report page and must judge metric coverage, category match, and unit compatibility. We propose **Orchestrator RegComAgent**, a few-shot agentic AI framework for multilingual ESG regulatory compliance verification. The system uses 12 held-out development examples to calibrate the boundary among YES, YES BUT NOT COMPLETE, and NO. Its workflow extracts page text, normalizes multilingual metric intent, searches evidence, verifies metric mention, category, unit, and completeness, and produces structured predictions through aggregation and selective review. A GPT 4o OrchestratorAgent coordinates GPT 4o-mini worker agents, while high-risk cases are reviewed by Claude 4.6 Sonnet and GPT 4o review agents. The framework is evaluated across six languages and compared with a multilingual MiniLM baseline and three few-shot single-core LLM systems. Orchestrator RegComAgent achieves the best overall performance, with the clearest gain on YES BUT NOT COMPLETE. The results suggest that orchestration, calibration, repair, and targeted review improve compliance verification, although partial disclosure, proxy evidence, unit mismatch, SASB index pages, and single-page constraints remain challenging. This study contributes an auditable orchestrator-controlled architecture for ESG compliance verification and supports ESG assurance and regulatory review.

Keywords: Agentic AI · Orchestrator Agent · Large Language Models · ESG Reporting · SASB · Compliance Verification · Multilingual NLP.

* Corresponding author.

1 Introduction

Over the past decade, Environmental, Social, and Governance considerations have evolved from a soft preference of responsible investors to a hard requirement of capital markets and regulators worldwide. The European Union has begun rolling out the Corporate Sustainability Reporting Directive, the International Sustainability Standards Board has promoted sustainability reporting standards, and regulators across Asia have increasingly required disclosures aligned with SASB or ISSB style metrics. A common feature of these mandates is that disclosure must be tied to a specific metric code, a prescribed disclosure category and a standardized unit of measure. ESG reports therefore often exceed one hundred pages, mix several languages, and combine semi structured tables with narrative text, making manual compliance review by regulators, third party assurance providers and ESG analysts laborious and inconsistent. Prior work on greenwashing and climate related disclosure has further shown that disclosure volume alone does not guarantee credibility [1,2], sharpening the need for scalable compliance verification.

The task setting of this study comes from the RegCom pilot shared task at NTCIR 19, which targets multilingual regulatory compliance checking for ESG reports [3]. We focus on Subtask 2. Given a SASB metric m and a single page p from an ESG report, the system must produce three outputs: `pred_label` $\in$ {YES, YES BUT NOT COMPLETE, NO}, indicating whether the page addresses the metric; `category_match`, indicating whether the page presentation matches the SASB category; and `unit_match`, indicating whether the unit of measure aligns with the SASB requirement [4]. Because each example provides only one page, the system must make fine grained judgements with limited context. The task therefore goes beyond ordinary topic classification and demands SASB domain knowledge, cross lingual semantic alignment and precise disclosure format reasoning.

The development of large language models has made zero shot and few shot domain tasks increasingly tractable. Brown et al. [5] show that large language models can perform many tasks without task specific training data, while Wei et al. [6] show that chain of thought prompting improves complex reasoning. These findings motivate the use of LLMs for structured judgement tasks, but a single prompt still leaves the system with weak control over evidence search, unit checking and completeness judgement. Agentic AI addresses this limitation by separating planning, action, verification and review into inspectable steps.

Preliminary experiments decomposed the judgement into five logical modules and compared several single core LLM reasoning systems, including Claude, GPT, Gemini and a multilingual MiniLM embedding baseline. Those results showed that agentic decomposition improves over embedding similarity, but also revealed persistent weakness on the YES BUT NOT COMPLETE class. The research objective of this study is therefore to design and evaluate an orchestrator controlled architecture in which a strong OrchestratorAgent functions as an AI project manager: it understands the overall compliance task, assigns narrower subtasks to AI workers, monitors risk signals and integrates their outputs into

final auditable predictions. Claude is incorporated as a conflict review agent, and GPT 4o is used as a final meta review agent for difficult cases.

This study asks whether an orchestrator controlled multi agent architecture can improve multilingual SASB single page compliance verification, whether a strong OrchestratorAgent can coordinate lower cost worker agents while preserving auditable intermediate outputs, and what kinds of errors remain after deterministic aggregation and targeted review. The contributions are threefold. We propose Orchestrator RegComAgent for multilingual ESG compliance verification; we report the final post reviewed result from the official experiment package; and we analyze the remaining errors by class, language, review transition, runtime trace and ESG topic. The remainder of the paper reviews related work in Section 2, describes the methodology and experimental design in Section 3, reports results and error analysis in Section 4, and concludes in Section 5.

2 Literature Review

2.1 Motivation: ESG Disclosure and Compliance Verification

ESG disclosure has become a default corporate practice, but the credibility and completeness of disclosed contents remain long standing concerns. Bingler et al. [1] train ClimateBERT on disclosures from TCFD supporting firms and conclude that voluntary support is often cheap talk, with firms selectively disclosing non material climate risk information; the open ClimateBERT release by Webersinke et al. [7] then provided a backbone for further domain adapted analyses. Kim and Yoon [2] show an analogous pattern at the fund level: signatories to the United Nations Principles for Responsible Investment attract significant inflows yet display no measurable improvement in fund level ESG scores. Mandatory disclosure standards such as SASB, ISSB and CSRD respond to this credibility problem by defining structured requirements, but they also create a verification problem: an analyst must determine whether a specific page or passage satisfies a specific metric definition, category and unit of measure.

The NTCIR 19 RegCom task is positioned at this verification gap. Unlike broad ESG topic classification, the task requires page level judgement against a metric definition, category and unit. It therefore benefits from systems that can separate evidence retrieval, semantic matching, category verification, unit checking and completeness judgement instead of treating the page as a single relevance classification input.

2.2 ESG NLP and Multilingual Sentence Embeddings

Early ESG NLP work focused on sentiment analysis, topic classification and evidence extraction. ClimateBERT and related work show that domain adapted language models can improve climate disclosure analysis [1,7], while later datasets such as ML Promise [8] and PromiseEval [9] move toward multilingual promise

verification in ESG reports. The RegCom task extends these settings by anchoring every judgement to a SASB metric and by requiring structured outputs beyond relevance detection.

Because the task spans Chinese, English, French, Japanese, Korean and Thai, multilingual sentence embeddings provide a natural lightweight baseline. Reimers and Gurevych [10,11] introduce Sentence BERT and multilingual knowledge distillation, producing models such as paraphrase multilingual MiniLM L12 v2. Such models map multilingual text into a shared semantic vector space and can be used for similarity based retrieval or classification. However, embedding similarity alone is limited for this task because topical similarity does not guarantee metric compliance, category match, unit match or completeness.

2.3 From Generative AI to Orchestrator Agentic AI

Large language models have evolved from single turn text generation toward agentic systems that plan, use tools, coordinate subtasks and integrate intermediate results. Cao et al. [12] survey the development of AI generated content and identify the move from content generation toward task executing intelligent agents as a major shift. Xi et al. [13] analyze LLM based agents through perception, planning, action and memory. Xie et al. [14] discuss large multimodal agents, which are relevant to ESG reports because many disclosures are embedded in tables and figures. Schneider [15] emphasizes controllability, verifiability and interoperability as key challenges in the move from generative to agentic AI. Empirical foundations for multi step reasoning include chain of thought prompting [6] and ReAct style reasoning [16].

Recent agent frameworks make the orchestration layer explicit. HuggingGPT treats an LLM as a controller that plans tasks, selects external expert models, executes subtasks and summarizes their outputs [17]. AutoGen generalizes this idea through configurable multi agent conversations in which agents can use LLMs, tools and human input [18]. MetaGPT further shows how role based collaboration and standardized operating procedures can make multi agent work resemble a structured software team [19]. More recent surveys emphasize communication architecture, scalability, security and multimodal integration in LLM based multi agent systems [20], while 2026 work highlights collaboration, failure attribution, self evolution and reliability limits as central issues for multi agent planning [21,22]. These studies suggest that performance depends not only on the strength of individual models but also on how roles, communication and review are coordinated.

Orchestrator RegComAgent follows this line of work. The OrchestratorAgent is not merely another classifier; it is the AI project manager of the system. It understands the page level SASB verification task, defines the control strategy, assigns worker agents to language normalization, evidence search, verification and writing, monitors risk signals, invokes review agents when needed and integrates final outputs. This distinction matters because ESG compliance verification requires both local evidence checking and global decision control.

Table 1. Distribution of the NTCIR 19 RegCom task 1 dataset by `pred_label` and report language.

Label	Total	%	Chinese	English	French	Japanese	Korean	Thai
YES	387	41.1%	92	75	93	38	43	46
YBNC	140	14.9%	13	34	21	30	33	9
NO	415	44.1%	84	84	60	54	67	66
Total	**942**	**100%**	189	193	174	122	143	121

Note. YBNC = *yes but not complete.*

3 Methodology and Experimental Design

3.1 Data and Statistics

The dataset comes from the NTCIR 19 RegCom shared task, covering ESG reports from listed companies across six languages: Chinese, English, French, Japanese, Korean and Thai. The reports span industries such as financial services, semiconductors, retail, energy, apparel and automotive. Each example is associated with one SASB metric and one ESG report page. Each row contains the report language, company or report identifier, SASB metric code, topic, metric description, page identifier, extracted page file name, SASB category, SASB unit of measure, key terms and a `what_counts` description.

The full task dataset contains 942 samples. In the final orchestrator experiment package, 11 demonstration or calibration samples are reserved before the official scoring run, and the final reported result is computed over 930 successfully evaluated instances after output validation. Table 1 summarizes the full dataset distribution before demonstration exclusion and run level filtering.

The NO class accounts for the largest share of the full dataset, reflecting how precise SASB metrics make many pages effectively non responsive. The YES BUT NOT COMPLETE class is the smallest class, but it has disproportionate influence on Macro F1 because it requires the system to identify semantically relevant evidence while also detecting missing sub items, incomplete scope, proxy disclosure or unit mismatch. English and Chinese are the largest language subsets, while Thai and Japanese are the smallest.

3.2 Task Formalisation and Evaluation Metrics

For each example, the system produces three structured fields: `pred_label`, `category_match` and `unit_match`. The label YES means that the page sufficiently discloses the required information for the target metric. The label YES BUT NOT COMPLETE means that the page mentions or partially addresses the metric, but the disclosure is incomplete, uses a proxy, misses required sub items, or does not fully satisfy the SASB requirement. The label NO means that the page does not address the metric sufficiently.

`category_match` checks whether the disclosure matches the SASB category, such as Quantitative or Discussion and Analysis. `unit_match` checks whether the reported unit of measure is compatible with the SASB requirement. When `pred_label` is NO, the auxiliary fields are normalized to N/A. Macro F1 and Micro F1 are used as the primary evaluation metrics. Macro F1 gives equal weight to the three labels and is important under class imbalance, especially because YES BUT NOT COMPLETE is the minority class. Micro F1 captures overall prediction consistency and is equivalent to accuracy in this single label setting, but we do not use accuracy as a primary metric.

Because all systems are evaluated on the same page metric instances, model comparison should be treated as paired rather than independent. When row level prediction files are available for two systems, the primary significance test is an exact McNemar test on paired correctness indicators, and a paired bootstrap over instances is used as a complementary check for Macro F1 differences. When only aggregate legacy scores are available, the comparison is reported descriptively rather than assigned a p value. This testing design is intended to distinguish genuine model or architecture differences from random variation caused by prompting, review agents or sampling.

3.3 Baseline

The baseline uses paraphrase multilingual MiniLM L12 v2 with rule based verification. Page text and the SASB metric description are encoded into multilingual sentence embeddings, and cosine similarity is used as a proxy for relevance. Rule based checks then estimate `category_match` and `unit_match`. This baseline is computationally efficient and provides a useful reference point, but it cannot reliably distinguish topical similarity from metric compliance.

The baseline also relies only on text extracted from the PDF and does not perform OCR. A non trivial fraction of ESG disclosures is rendered as tables, charts or figures, especially for greenhouse gas emissions, workforce diversity, energy consumption and financed emissions. PDF text extraction alone may therefore lose access to information that is visible on the page image. This caveat is important when interpreting the gap between the MiniLM baseline and Orchestrator RegComAgent.

3.4 Single Core LLM Comparative Systems

In addition to the MiniLM baseline, we keep three preliminary single core LLM systems as comparison points: Claude 4.6 Sonnet, GPT 5.5 high and Gemini 3 Pro. These systems use the same task inputs and the same output schema, but they do not implement the orchestrator controlled worker structure. Each system performs the page level judgement through a single reasoning core with a structured prompt. They are therefore not Orchestrator RegComAgent variants; they are single core LLM comparison systems used to separate the effect of model capability from the effect of orchestration.

This distinction is important for interpreting the results. A strong single model may produce good local reasoning, but it has no explicit project manager layer for assigning work, monitoring risk signals, applying deterministic repair or invoking targeted review. Orchestrator RegComAgent is evaluated as a system architecture, not merely as another model choice.

3.5 The Orchestrator RegComAgent Framework

Inputs and outputs. Orchestrator RegComAgent receives two inputs. The first input is the SASB metric row, including `metric_description`, `sasb_category`, `sasb_unit_of_measure`, `sasb_key_terms`, and `sasb_what_counts`. The second input is a single ESG report PDF page. Before the agent reasoning process begins, the system extracts the page text from the PDF page using a deterministic PDF text extraction step. The final output consists of three structured fields: `pred_label` $\in$ {YES, YES BUT NOT COMPLETE, NO}, `category_match` $\in$ {YES, NO, N/A}, and `unit_match` $\in$ {YES, NO, N/A}. The system also preserves intermediate traces and timing logs so that each prediction can be inspected after inference.

Figure 1 sketches the overall architecture. Orchestrator RegComAgent is implemented as an orchestrator-controlled sequential multi-agent pipeline rather than a parallel debate system. The GPT-4o OrchestratorAgent acts as the central coordinator of the entire workflow. At every stage, the OrchestratorAgent actively communicates with each downstream worker agent: it dispatches task-specific instructions, receives intermediate outputs, evaluates whether the results satisfy the mandatory checks and risk criteria it has defined, and decides whether to proceed, request revision, or escalate to a review agent. After all worker agents and review agents have returned their outputs, the OrchestratorAgent integrates the intermediate results and produces the final consolidated prediction. In this sense, no final label is emitted without passing through the OrchestratorAgent's integration layer.

Stage 1: Planning, Normalization, and Evidence Search. The OrchestratorAgent first analyzes the task, defines the overall compliance strategy, identifies mandatory sub-item checks, records risk notes, and generates agent-specific instructions. It then dispatches these instructions to a GPT-4o-mini TaskPlanner, which converts the high-level strategy into concrete worker steps and execution order. The OrchestratorAgent next interacts with a GPT-4o-mini LanguageNormalizationAgent, directing it to align the English SASB metric intent, key terms, and unit aliases with the page language. This step is critical because the task covers multilingual ESG reports and the same SASB concept may be expressed differently across Chinese, English, French, Japanese, Korean, and Thai pages. The OrchestratorAgent then instructs a GPT-4o-mini SearchAgent to retrieve positive and negative evidence from the extracted page text, and receives a structured evidence representation that is forwarded to the downstream verification stage.

Stage 2: Verification and Structured Writing. The OrchestratorAgent passes the structured evidence and its mandatory check list to a GPT-4o-mini VerifyAgent, which examines four decision dimensions: whether the exact SASB metric or submetric is semantically mentioned, whether the disclosure category matches the SASB requirement, whether the reported unit of measure is compatible with the required unit, and whether the disclosure is complete, partial, or absent. The OrchestratorAgent then forwards the verification results to a GPT-4o-mini WriterAgent, which generates a draft prediction containing the three required fields: `pred_label`, `category_match`, and `unit_match`. The draft is returned to the OrchestratorAgent for evaluation before proceeding to the aggregation stage.

Stage 3: Aggregation and Review. The OrchestratorAgent passes the draft prediction to a deterministic Result Aggregator, which validates the output schema, normalizes invalid labels, enforces consistency between `pred_label` and the auxiliary fields, and applies rule-based overrides for known boundary cases. For example, when `pred_label` is NO, the system normalizes `category_match` and `unit_match` to N/A. The aggregator also handles rule-based corrections for specific metric patterns, unit interpretation issues, SASB index pages, and common false-positive or false-negative conditions. The aggregated result is returned to the OrchestratorAgent, which evaluates whether targeted review is required.

For high-risk rows, the OrchestratorAgent dispatches the current prediction, SearchAgent evidence, VerifyAgent judgement, high-risk reasons, and the single-page PDF text to the Claude ConflictReviewAgent (Claude 4.6 Sonnet). High-risk triggers include boundary predictions such as YES BUT NOT COMPLETE, unit mismatch, category mismatch, Discussion and Analysis metrics predicted as NO, multilingual NO predictions when enabled, short PDF text, and SASB index page patterns. Claude 4.6 Sonnet returns a revised structured prediction, which the OrchestratorAgent receives and records.

Finally, for rows flagged as disagreement or high-risk after the Claude review stage, the OrchestratorAgent submits the full evidence package, including the current prediction, the base repaired prediction, the Claude review result, worker evidence, verification judgement, and the single PDF page text, to a GPT-4o MetaReviewAgent for final arbitration. The MetaReviewAgent's decision is returned to the OrchestratorAgent, which normalizes and integrates it into the final consolidated CSV output.

In this design, every intermediate result flows through the OrchestratorAgent, which functions as both a task-level coordinator and the integration point for all agent outputs. Lower-cost worker agents perform planning, language normalization, evidence search, verification, and structured writing under the OrchestratorAgent's direction. Deterministic aggregation ensures schema validity and rule consistency, the Claude ConflictReviewAgent provides targeted conflict review, and the GPT-4o MetaReviewAgent performs final arbitration for selected difficult cases. Because no label is finalized without the OrchestratorAgent's integration step, the architecture preserves auditable intermediate outputs while combining LLM-based reasoning with deterministic post-processing and selective expert review.

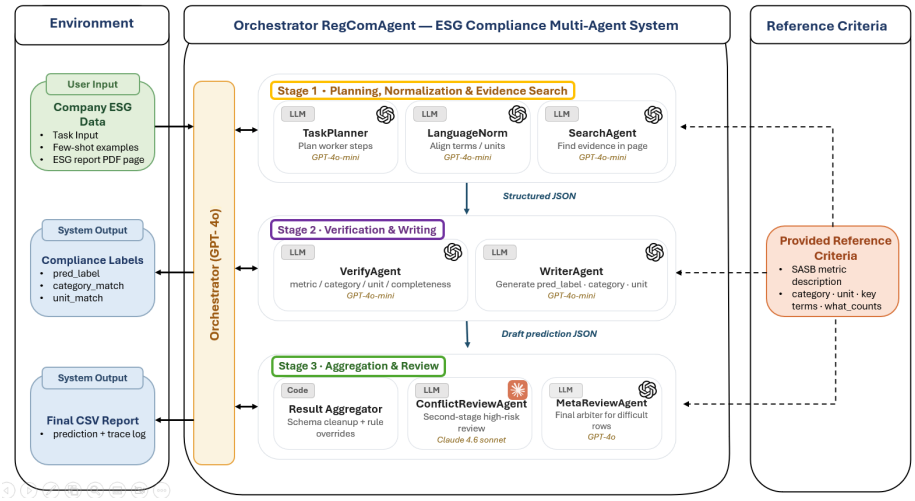

Fig. 1. Orchestrator RegComAgent multi-agent architecture.

4 Results and Analysis

4.1 Overall Performance

Table 3 reports Macro F1 and Micro F1 for Orchestrator RegComAgent and the comparison systems. Orchestrator RegComAgent achieves Macro F1 = 0.6312 and Micro F1 = 0.6860, with 638 correct predictions out of 930 evaluated instances. This improves over the strongest single core comparison system, Claude 4.6 Sonnet, which reached Macro F1 = 0.582 and Micro F1 = 0.664. The improvement is most meaningful in Macro F1 because Macro F1 is sensitive to the minority YES BUT NOT COMPLETE class.

The baseline does not perform OCR or use image input, so part of its gap relative to Orchestrator RegComAgent likely comes from missing table and chart evidence. Nevertheless, the size of the improvement also indicates that structured reasoning, deterministic aggregation and targeted review help with judgement dimensions that embedding similarity does not represent. After aligning the available row level prediction files, we further test whether the differences between Orchestrator RegComAgent and the comparison systems are statistically reliable using paired t-tests on per-instance correctness scores. Table 4 reports the mean scores, difference, t-statistic and p-value for each comparison. Positive differences indicate that Orchestrator RegComAgent outperforms the comparison system on the common aligned rows.

The paired tests support the model level pattern in Table 3. Orchestrator RegComAgent produces more cases that are correct only under the orchestrator system than cases that are correct only under each comparison system, including the strongest single core Claude system. The smaller common row count for Claude reflects missing or non alignable rows in the supplied batch outputs. The

Table 2. Experimental configuration comparison for the MiniLM baseline, single-core LLM systems, and Orchestrator RegComAgent.

Experiment	Reasoning core	Auxiliary component
Exp 1	paraphrase multilingual MiniLM L12 v2	Cosine similarity and rule based category and unit checks
Exp 2	Claude 4.6 Sonnet	Single core structured prompt comparison system
Exp 3	GPT 5.5 high	Single core structured prompt comparison system
Exp 4	Gemini 3 Pro	Single core structured prompt comparison system
Exp 5	Orchestrator RegComAgent	GPT 4o OrchestratorAgent, GPT 4o mini workers, deterministic aggregation, Claude conflict review and GPT 4o meta review

Table 3. Overall performance of Orchestrator RegComAgent and comparison systems.

Configuration	Macro F1	Micro F1
Orchestrator RegComAgent with Claude and GPT 4o review	**0.6312**	**0.6860**
Single core Claude 4.6 Sonnet	0.582	0.664
Single core GPT 5.5 high	0.567	0.616
Single core Gemini 3 Pro	0.483	0.568
Baseline: paraphrase multilingual MiniLM L12 v2	0.437	0.501

paired t-tests support the model level pattern in Table 3. Orchestrator RegComAgent produces significantly more correct predictions than each comparison system, including the strongest single core Claude system ($t = 2.329$, $p = .020$). The smaller common row count for Claude reflects missing or non-alignable rows in the supplied batch outputs.

4.2 Per Class F1

Table 5 reports F1 by `pred_label` class. Orchestrator RegComAgent obtains the best F1 for YES and YES BUT NOT COMPLETE, while the single core Claude configuration remains strongest on NO. The gain on YES BUT NOT COMPLETE is especially important because this class was the major bottleneck in the preliminary systems.

The final prediction distribution is 357 predictions of YES, 146 predictions of YES BUT NOT COMPLETE and 427 predictions of NO. The corresponding gold support in the evaluated set is 376, 141 and 413 respectively, showing that the final system is not simply biased toward the majority class.

Table 4. Comparison of row-level correctness between Orchestrator RegComAgent and the baseline systems using paired significance tests.

Comparison system	n	Orchestrator	Comparison	Δ Micro F1	t	p
Single core Claude 4.6 Sonnet	914	0.684	0.644	+0.039	2.329	$.020^{*}$
Single core GPT 5.5 high	930	0.682	0.624	+0.058	3.568	$< .001^{***}$
Single core Gemini 3 Pro or Lite	930	0.682	0.576	+0.105	5.775	$< .001^{***}$
Baseline MiniLM	930	0.682	0.496	+0.186	9.366	$< .001^{***}$

Note. Scores are paired per-instance correctness means, equivalent to Micro F1 in this single-label setting. Δ Micro F1 denotes Orchestrator RegComAgent minus the comparison system. $^{*}p < .05$; $^{***}p < .001$.

Table 5. Comparison of per-class F1 between Orchestrator RegComAgent and the baseline systems for `pred_label`.

Class	Orchestrator RegComAgent	Claude 4.6	GPT 5.5	Gemini 3	Baseline
YES	**0.7067**	0.631	0.642	0.498	0.537
YBNC	**0.4321**	0.327	0.333	0.270	0.220
NO	0.7548	**0.789**	0.725	0.682	0.553

Note. YBNC = *yes but not complete*. Higher F1 indicates better class-specific prediction quality.

4.3 Error Type Analysis

Table 6 reports the major error types across Orchestrator RegComAgent and the comparison systems.

The final confusion pattern shows that boundary control remains difficult. True YES predicted as NO occurs when valid evidence is qualitative, indirect or phrased differently from the SASB metric. True NO predicted as YES occurs when the system accepts topically related ESG content that does not satisfy the precise SASB requirement. True YES predicted as YES BUT NOT COMPLETE reflects conservative completeness or unit interpretation. The two error directions involving YES BUT NOT COMPLETE show the central tension of the task: the system must decide whether evidence is absent, partial or complete.

4.4 Cross Lingual Consistency, Review Stage and Failure Mode Summary

Remaining wrong predictions are distributed across all six languages: English 68, French 68, Japanese 44, Chinese 39, Thai 39 and Korean 34. The distribution shows that errors are not confined to one language. English and French contain the largest number of remaining errors, partly because their evaluated sample counts and topic composition are larger in the target set. Japanese, Thai, Chinese and Korean remain challenging because translated ESG terminology and local reporting styles can differ substantially from English SASB metric descriptions.

The most difficult remaining topics are Greenhouse Gas Emissions with 26 wrong predictions, Financed Emissions with 21, Labour Conditions in the Supply Chain with 14, Financial Inclusion & Capacity Building with 12, Data Security with 12, Activity Metrics with 11, Incorporation of ESG Factors in Investment Management & Advisory with 10, and Materials Efficiency & Recycling with

Table 6. Comparison of major `pred_label` error transitions between Orchestrator RegComAgent and the baseline systems.

Configuration	Y→N	Y→YBNC	YBNC→Y	N→Y
Orchestrator RegComAgent	69	48	38	60
Single core Claude 4.6 Sonnet	89	62	47	28
Single core GPT 5.5 high	66	87	51	42
Single core Gemini 3 Pro	178	38	43	45
Baseline	117	38	78	150

Note. Each column reports a true-label → predicted-label transition, e.g., Y→N means gold YES predicted as NO. Y = YES; N = NO; YBNC = *yes but not complete*. These transitions help analyze false negatives, over-conservative partial labels, partial-to-complete errors, and false positives.

9. Greenhouse gas and financed emissions remain difficult because they often require multiple scopes, percentages, intensity measures or financed emissions categories. Labour and supply chain metrics often contain narrative descriptions that are relevant but incomplete. Data security metrics are difficult because generic cybersecurity policies can be topically similar to the SASB requirement but fail to disclose required incident counts or corrective actions.

The GPT 4o meta review stage reviewed 230 difficult rows. The observed review trace transitions were 54 from YES to YES BUT NOT COMPLETE, 23 from YES to NO, 21 from NO to YES BUT NOT COMPLETE, 2 from YES BUT NOT COMPLETE to NO, and 1 from YES BUT NOT COMPLETE to YES. The most frequent transition confirms that the review stage primarily improves completeness judgement. The second important transition, from NO to YES BUT NOT COMPLETE, shows that review can recover partial evidence missed by the base pipeline.

4.5 Agent Work Logic and Trace Case Studies

The trace files allow us to examine how Orchestrator RegComAgent reaches its decisions, rather than only whether its final predictions are correct. Three representative cases illustrate the system logic.

In a supplier environmental assessment case, the OrchestratorAgent converted the SASB requirement into mandatory checks: supplier assessment percentages, separation of Tier 1 and beyond Tier 1 suppliers, and completion of Higg FEM or an equivalent assessment. The TaskPlanner turned these checks into search and verification steps. Since the SearchAgent found no tier-specific quantitative evidence, the VerifyAgent returned NO. This shows that topic-relevant supply chain language is insufficient when the metric requires disaggregated disclosure.

A second case concerns priority raw materials. Although the page mentioned traceability and internal sourcing standards, the OrchestratorAgent required the workers to verify purchase amounts and certified amounts or percentages by standard. Because the page did not report these quantities in the required form,

the VerifyAgent rejected a full YES. This case shows that semantic relevance does not necessarily imply complete compliance.

A third case comes from the meta-review traces. Some NO predictions were changed to YES BUT NOT COMPLETE when partial qualitative evidence was found, such as ESG integration methods, chemical risk controls, financed emissions estimation methods, or supply chain screening. Conversely, some YES predictions were downgraded when the page provided only percentages without absolute quantities, total values without disaggregation, or brief policy statements without procedural details. These transitions show that review agents mainly refine the boundary among absent, partial, and complete disclosure.

The traces also reveal a deployment limitation: because exact token counts were not logged, token use cannot yet be compared across agents. Future runs should record prompt and completion tokens for every agent call and preserve row-level predictions for all baselines to support paired significance testing.

Overall, the remaining failure modes include partial-disclosure boundary errors, multilingual semantic mismatch, SASB index or table ambiguity, unit and category mismatch, and the single-page constraint. Future systems should therefore combine explicit sub-item checklists, unit conversion tools, full standard retrieval, OCR or multimodal page understanding, and cross-page report retrieval.

5 Conclusion

This study proposes Orchestrator RegComAgent for the RegCom shared task, shifting the main contribution from single-core LLM comparison to an orchestrator-controlled Agentic AI architecture for multilingual ESG compliance verification. The final system combines a strong OrchestratorAgent, lower-cost worker agents, a deterministic Result Aggregator for schema validation and repair, a conflict review agent, and a final meta-review agent. This design allows the system to separate task planning, multilingual normalization, evidence search, verification, structured writing, aggregation, and review while keeping intermediate decisions inspectable.

Orchestrator RegComAgent outperforms the single-core LLM comparison systems and the MiniLM baseline, showing that orchestrator-controlled decomposition and targeted review can improve page-level ESG compliance verification. The strongest improvement appears on the YES BUT NOT COMPLETE class, confirming that partial disclosure should be treated as a distinct compliance state rather than a middle confidence label. This result suggests that the key challenge is not only identifying whether a page is topically related to a SASB metric, but also determining whether the evidence is complete, properly categorized, and expressed in a compatible unit.

The main contribution of this study is an auditable multi-agent framework that operationalizes ESG compliance checking as a coordinated reasoning process rather than a single-step classification task. By assigning different agents to planning, language normalization, evidence search, verification, writing, and review, Orchestrator RegComAgent makes the decision process more transparent

and easier to diagnose. The framework also demonstrates how deterministic repair and selective review can be combined with LLM reasoning to reduce schema errors, improve boundary decisions, and provide traceable evidence for each final label.

The academic implication is that ESG compliance verification should be studied not only as model selection, but also as orchestration design. The results indicate that system-level coordination, role separation, and review mechanisms can be as important as the choice of the underlying LLM. The practical implication is that regulators, assurance providers, and ESG analysts can use such systems for page-level triage, especially to identify partial disclosures that require human attention. The managerial implication is that firms can treat system traces as governance artifacts: evidence gaps, unit mismatches, category conflicts, review triggers, and unresolved boundary cases become observable signals for improving future disclosure quality.

The study has several limitations. The final run still judges only one page at a time, so cross-page evidence cannot be used. The MiniLM baseline does not integrate OCR, which means that the baseline comparison partly reflects different access to visual evidence. The final review stage also depends on closed model APIs, leaving open-source model families for future investigation. Future work will extend Orchestrator RegComAgent with retrieval over SASB standard documents, explicit sub-item verification, unit conversion tools, OCR and multimodal page understanding, cross-page report retrieval for full report analysis, and systematic paired significance testing with confidence intervals for all model comparisons.

Acknowledgments. This research was supported by the Industrial Technology Research Institute (ITRI) and National Taipei University (NTPU), Taiwan, under grants NTPU 114A513E01 and NTPU 113A513E01; the National Science and Technology Council (NSTC), Taiwan, under grant NSTC 114 2425 H 305 003; and National Taipei University (NTPU) under grant 114 NTPU_ORDA F 004.

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Bingler, J. A., Kraus, M., Leippold, M., Webersinke, N.: Cheap Talk and Cherry Picking: What ClimateBert has to say on Corporate Climate Risk Disclosures. Finance Research Letters **47**, 102776 (2022)
2. Kim, S., Yoon, A.: Analyzing Active Fund Managers' Commitment to ESG: Evidence from the United Nations Principles for Responsible Investment. Management Science **69**(2), 741 to 758 (2023)
3. NTCIR 19 RegCom Organizers: Task Description of the NTCIR 19 RegCom Task: Multilingual Regulatory Compliance Checking for ESG Reports (2025)
4. Sustainability Accounting Standards Board: SASB Standards (2023)
5. Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., et al.: Language Models are Few Shot Learners. In: NeurIPS 2020 (2020)

6. Wei, J., Wang, X., Schuurmans, D., Bosma, M., Ichter, B., Xia, F., Chi, E., Le, Q., Zhou, D.: Chain of Thought Prompting Elicits Reasoning in Large Language Models. In: NeurIPS 2022 (2022)
7. Webersinke, N., Kraus, M., Bingler, J. A., Leippold, M.: ClimateBERT: A Pretrained Language Model for Climate Related Text. arXiv:2110.12010 (2022)
8. Seki, Y., Shu, H., Lhuissier, A., Lee, H., Kang, J., Day, M. Y., Chen, C. C.: ML Promise: A Multilingual Dataset for Corporate Promise Verification. arXiv:2411.04473 (2024)
9. Chen, C. C., Seki, Y., Shu, H., Lhuissier, A., Kang, J., Lee, H., Day, M. Y., Takamura, H.: SemEval 2025 Task 6: Multinational, Multilingual, Multi Industry Promise Verification. In: SemEval 2025, pp. 2461 to 2471 (2025)
10. Reimers, N., Gurevych, I.: Sentence BERT: Sentence Embeddings Using Siamese BERT Networks. In: EMNLP 2019, pp. 3982 to 3992 (2019)
11. Reimers, N., Gurevych, I.: Making Monolingual Sentence Embeddings Multilingual using Knowledge Distillation. In: EMNLP 2020 (2020)
12. Cao, Y., Li, S., Liu, Y., Yan, Z., Dai, Y., Yu, P. S., Sun, L.: A Comprehensive Survey of AI Generated Content (AIGC): A History of Generative AI from GAN to ChatGPT. arXiv:2303.04226 (2023)
13. Xi, Z., Chen, W., Guo, X., He, W., Ding, Y., Hong, B., et al.: The Rise and Potential of Large Language Model Based Agents: A Survey. arXiv:2309.07864 (2023)
14. Xie, J., Chen, Z., Zhang, R., Wan, X., Li, G.: Large Multimodal Agents: A Survey. arXiv:2402.15116 (2024)
15. Schneider, J.: Generative to Agentic AI: Survey, Conceptualization, and Challenges. arXiv:2504.18875 (2025)
16. Yao, S., Zhao, J., Yu, D., Du, N., Shafran, I., Narasimhan, K., Cao, Y.: ReAct: Synergizing Reasoning and Acting in Language Models. In: ICLR 2023 (2023)
17. Shen, Y., Song, K., Tan, X., Li, D., Lu, W., Zhuang, Y.: HuggingGPT: Solving AI Tasks with ChatGPT and its Friends in Hugging Face. In: NeurIPS 2023 (2023)
18. Wu, Q., Bansal, G., Zhang, J., Wu, Y., Li, B., Zhu, E., Jiang, L., et al.: AutoGen: Enabling Next Gen LLM Applications via Multi Agent Conversation. arXiv:2308.08155 (2023)
19. Hong, S., Zhuge, M., Chen, J., Zheng, X., Cheng, Y., Wang, C., et al.: MetaGPT: Meta Programming for A Multi Agent Collaborative Framework. In: ICLR 2024 (2024)
20. Yan, B., Zhang, X., Zhang, L., Zhang, L., Zhou, Z., Miao, D., Li, C.: Beyond Self Talk: A Communication Centric Survey of LLM Based Multi Agent Systems. arXiv:2502.14321 (2025)
21. Qi, S., Ma, J., Xing, R., Guo, W., Huang, X., Gao, Z., et al.: Beyond Individual Intelligence: Surveying Collaboration, Failure Attribution, and Self Evolution in LLM Based Multi Agent Systems. arXiv:2605.14892 (2026)
22. Ao, R., Gao, S., Simchi Levi, D.: On the Reliability Limits of LLM Based Multi Agent Planning. arXiv:2603.26993 (2026)
23. Mehra, S., Louka, R., Zhang, Y.: Textual Evidence Extraction for ESG Scores. In: Proceedings of the 4th Workshop on Financial Technology and Natural Language Processing (2022)