RETUYT-INCO at BEA 2025 Shared Task: How Far Can Lightweight Models Go in AI-powered Tutor Evaluation?

Santiago Góngora † and **Ignacio Sastre** † and **Santiago Robaina**
Ignacio Remersaro and **Luis Chiruzzo** and **Aiala Rosá**

Instituto de Computación, Facultad de Ingeniería, Universidad de la República
Montevideo, Uruguay

Abstract

In this paper, we present the RETUYT-INCO participation at the BEA 2025 shared task. Our participation was characterized by the decision of using relatively small models, with fewer than 1B parameters. This self-imposed restriction tries to represent the conditions in which many research labs or institutions are in the Global South, where computational power is not easily accessible due to its prohibitive cost. Even under this restrictive self-imposed setting, our models managed to stay competitive with the rest of teams that participated in the shared task. According to the $exact\ F_1$ scores published by the organizers, the performance gaps between our models and the winners were as follows: 6.46 in *Track 1*; 10.24 in *Track 2*; 7.85 in *Track 3*; 9.56 in *Track 4*; and 13.13 in *Track 5*. Considering that the minimum difference with a winner team is 6.46 points — and the maximum difference is 13.13 — according to the $exact\ F_1$ score, we find that models with a size smaller than 1B parameters are competitive for these tasks, all of which can be run on computers with a low-budget GPU or even without a GPU.

1 Introduction

The remarkable advances in the development of Large Language Models (LLMs) in recent years have turned Natural Language Processing into a discipline with great potential for application in different domains, and *Education* is not the exception (Ignat et al., 2024). However, these technological advances are not affordable to everyone. The cost of closed models — which are the most powerful and are typically considered the State-of-the-art in NLP — and the expensive infrastructure required to use large open models, coupled with negative effects on the environment, make research on other methods still essential.

Our RETUYT-INCO team, as a research lab from South America, is no exception to this reality. Naturally, we are concerned about these issues and, consequently, we have focused on experimenting with open models in recent editions of the BEA shared tasks. For the 2023 shared task, consisting in generating teacher responses in educational dialogues (Tack et al., 2023), we participated using open models, obtaining competitive results (Baladón et al., 2023). One of the highlights of our participation was the *"Hello" baseline*, a simple strategy we followed which achieved remarkable results, unveiling the fragility of BERTScore (Zhang et al., 2020). More recently, for the 2024 BEA shared task, consisting in performing simplification experiments for different languages (Shardlow et al., 2024), we mainly focused on fine-tuning BERT and Mistral models (i.e., open models), even using synthetic data in some cases (Sastre et al., 2024).

In this paper, we present the RETUYT-INCO participation in the **five tracks** of the BEA 2025 Shared Task: *Pedagogical Ability Assessment of AI-powered Tutors* (Kochmar et al., 2025). This year, in addition to maintaining our restriction of working with open models, we challenged ourselves with an extra restriction: to experiment only with language models of fewer than a billion parameters and classical machine learning (ML) approaches. We will call these *lightweight* models, as they have to be small enough to run on a low-end GPU or with no GPU at all. This restriction is related to the situation many research labs face every day in the Global South: the lack of minimum resources to run what other regions consider *small* models (7B parameters or more). In our case, we have limited access to a national computing cluster, which we can use to fine-tune LLMs up to 7B parameters, but we do not have resources to host the fine-tuned

† These (corresponding) authors contributed equally to this work: {sgongora,isastre}@fing.edu.uy

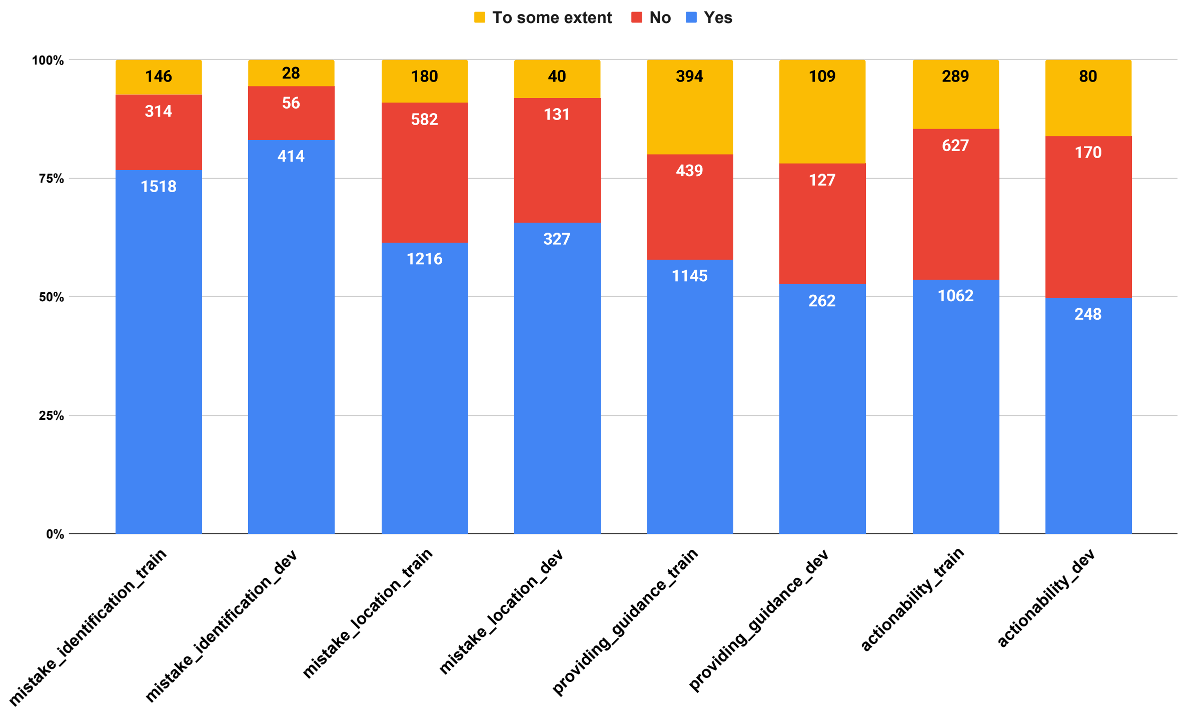

Figure 1: Class balance in our `train` and `dev` sets. The columns are coupled according to classes.

LLMs and use them in real applications.

Moreover, this is not the only motivation for this self-imposed restriction, as one of the research lines of our lab is the application of NLP tools to aid teachers in rural areas (Chiruzzo et al., 2022; Rosá et al., 2025). In such contexts it is very unlikely to use state-of-the-art LLMs, due to the impossibility of using them through APIs (since children's privacy is key, sending private data to third-party servers is not an option), and the prohibitive cost of installing capable GPUs in trustworthy servers.

Overall, throughout this paper, we will try to answer the general research question that motivated our participation: *What is the performance gap between lightweight models and those state-of-the-art models, which would naturally have a better chance of winning the competition?*.

2 Dataset

For this edition, the dataset consists of 300 conversations (Maurya et al., 2025). Each dialogue is composed of interactions between a teacher and a math student. In the final turn of each dialogue the student shows clear confusion about a concept, and the dataset includes potential tutor *responses* intended to help the student. These responses — some of them generated by seven LLM-based tutors and others written by human tutors — are also evaluated by human evaluators (using **Yes**, **No** or **To some extent**) according to four dimensions of interest that coincide with the four proposed tracks in the shared-task: *mistake identification*, *mistake location*, *providing guidance* and *actionability*.

Due to the lack of a specific *development* set, during the first month we split the official dataset published by the organizers into two parts: 80% We decided to do this split focusing on the conversations — and not on the responses — trying to ensure that each conversation and all its responses stayed either in the `train` or the `dev` set. As a consequence, our *train-dev* split may not preserve the class balance of the original set. Figure 1 shows the class balance for each dimension in our `train` and `dev` partition.

3 Considered approaches

For our experiments we considered classical ML classification algorithms, BERT-based approaches and fine-tuning a small autoregressive language model. Since all the tracks in the shared task are classification problems, many of the models we considered were used in more than one track. All of them were trained (or fine-tuned) using our `train` set, running on GoogleColab[1] or a national computational cluster (see Section 3.4.1). At the end of this section — in Subsection 3.5 — we will show

[1]https://colab.research.google.com/

the results we obtained when evaluating them on the dev set.

3.1 Preliminary experiments

To gain a greater understanding of how challenging the *tracks* were, we performed four preliminary experiments with increasing degree of complexity.

The first and most basic one consisted in answering always **yes**, what naturally degraded the F_1 $macro$ score, since the **No** and **To some extent** classes were never chosen. Then, we tried using a random classifier, which consistently yielded accuracy values around 33%

Additionally, before imposing ourselves the constraint of using *lightweight* models only, we wanted to have an informed perception of how well bigger LLMs could perform in these tracks. Therefore, we explored both prompting a closed model via an API and fine-tuning an open model. For prompting, the model we chose was `Gemini Flash 2.0 Lite`[2], using the prompt reported in the paper that presented the dataset (Maurya et al., 2025). For fine-tuning, we chose `Llama 3.1 8B Instruct` (AI@Meta, 2024), and used Low Rank Adaptation (LoRA) (Hu et al., 2022). This experiment follows the same methodology explained in Section 3.4.

3.2 Classical Machine Learning approaches

Among all the available algorithms in Sklearn[3] we tried, those that had the best performance on the dev set were *Random Forest*, *SVC* (Support-vector classifier) and *k-NN* (k-Nearest Neighbors). To represent the input texts we experimented with Bag of Words and TF-IDF, trying different n-gram ranges from $n = 1$ to $n = 8$.

Since all these algorithms have problems capturing the complexities of long-context dependencies, after some experimentation we decided to train the models using the *response* text only, i.e. without taking into consideration the full interaction between the student and the tutor. Those preliminary experiments we did with the full interaction (i.e. concatenating the response of the tutor to the conversation history) had a notably lower performance.

3.3 BERT-based approaches

We also tried BERT-based approaches. We experimented with fine-tuning them, combining them with classification algorithms and also with some rules.

3.3.1 BERT for Tracks 1–4

We implemented a simple method by fine-tuning a simple BERT model for tracks 1 through 4. Specifically, we finetuned the DistilBERT `distilbert-base-uncased` variant (Sanh et al., 2020), a compact and computationally efficient distillation of BERT with approximately 66 million parameters.

For this experiment, we only considered the response text as input data, without the conversation history. We fine-tuned the model to predict each of the target variables (`mistake_identification`, `mistake_location`, `providing_guidance`, `actionability`). We initially tried to fine-tune the model in a three-class configuration, but our experiments were unable to predict any value of the class **To some extent** whatsoever, so we changed the approach. We ended up training two-class models, joining **No** and **To some extent** as the negative class. After fine-tuning, we analyzed the logit of the positive class and observed that even if both classes were lumped together during training, the **No** values actually got lower logit than the **To some extent** values, which allowed us to define thresholds to separate the three classes.

The hyperparameters in these experiments were the number of training epochs (from 1 to 3) and two thresholds to distinguish the frontier between **No** and **To some extent**, and between **To some extent** and **Yes**, which depending on the target output could vary between -1 and +1. In this round of experiments, we used Adam optimization with a learning rate of 5×10^{-6}.

3.3.2 BERT for Track 5

In our approach to track 5, the objective was to classify the tutor identity based once again solely on the provided response text. For fine-tuning, the following parameters were used: a learning rate of 2×10^{-5}, a weight decay of 0.01, a training duration of 4 epochs, and batch sizes set to 16.

3.3.3 Sentence Embeddings

In addition to fine-tuning, we explored the use of BERT-like models to generate sentence embeddings (Reimers and Gurevych, 2019), which were then combined with classical ML methods for classification.

For tracks 1–4, we used the

[2] `https://deepmind.google/technologies/gemini/flash-lite/`

[3] https://scikit-learn.org/stable/supervised_learning.html

`multilingual-e5-large-instruct`[4] model (Wang et al., 2024), a multilingual encoder initialized from `xlm-roberta-large` (Conneau et al., 2019) (561M parameters). We generated a sentence embedding for each example in our training partition and then used those embeddings as input to classical classifiers: k-NN and multilayer perceptron (MLP). For this approach we explored three different input configurations:

- **Response-only:** The input consists solely of the embedding corresponding to the response to be evaluated.
- **Response + conversation history:** The input is formed by concatenating the embedding of the response with the embedding of the full conversation history.
- **Response + conversation history + LLM probabilities:** The input extends the previous configuration by appending the probabilities assigned to the three class labels by the fine-tuned LLM (see Section 3.4).

For the k-NN classifier, we chose $k = 9$ based on the performance on our dev set prior to submitting predictions for the competition's test set. For the MLP, we used a simple model with no hidden layer and trained it until convergence, defined as no improvement greater than a tolerance of 1×10^{-4} for 10 consecutive iterations.

For the mistake identification dimension, due to the high class imbalance in the training data, we applied under-sampling by fitting the k-NN classifier on a perfectly balanced subset. This subset contained an equal number of examples for each class, matching the count of the least frequent class. As shown in Section 4, this strategy led to improved performance. For this classifier, different values of k for each track were chosen (mistake identification: 415; mistake location: 540; providing guidance: 125; actionability: 96).

For track 5, we explored leveraging the DistilBERT model that was previously fine-tuned for direct sequence classification (as described in the previous section). In this setup, the core transformer layers (the base model, without the classification head) of this fine-tuned DistilBERT were employed as a feature extractor. Embeddings were generated for the "response" texts. These DistilBERT-derived embeddings were used as input features for an XGBoost classifier (Chen and Guestrin, 2016), which was configured for multiclass classification corresponding to the number of tutor labels.

3.3.4 BERT approach + Educated guess

Another experimental approach we tried for track 5 was to take the predictions of the BERT + XGBoost model and, based on the distribution of the predicted labels, guess some tutor identities. Under the premise that if the model predicted correctly the majority of the time the correct tutor for a certain label, then taking the same prediction for a label might improve the performance of the model. Therefore, for this approach we modified the predictions of BERT + XGBoost forcing to always classify Tutor9 as “Novice”, Tutor2 as “Mistral” and Tutor3 as “Llama31405B”.

Unfortunately, this approach turned out to perform poorly in comparison to the BERT + XGBoost one, denoting that the labels shown in the test dataset might not have a direct mapping with the actual classes.

3.4 Fine-tuning autoregressive LM

In addition to using encoder-only transformers such as BERT, we also experimented with decoder-only LMs. For these experiments, we only focused on the first four tracks. Although these tracks are framed as classification and are therefore usually better suited to encoder-only architectures, we wanted to compare BERT-style fine-tuned models with similarly sized, fine-tuned autoregressive LMs. Specifically, we used `Qwen2.5-0.5B-Instruct`[5] (Team, 2024; Yang et al., 2024), which has 494 million parameters and has undergone instruction tuning.

3.4.1 Training

We performed full fine-tuning on our train partition. Each example was converted into a prompt following the prompt template adopted during the model's instruction tuning phase. The prompt (available in Appendix A) consists of:

- **System prompt:** We used the same system prompt reported in the shared-task dataset paper (Maurya et al., 2025), which was also used for LLM-based evaluation.

[4] `https://huggingface.co/intfloat/multilingual-e5-large-instruct`

[5] `https://huggingface.co/Qwen/Qwen2.5-0.5B-Instruct`

- **User message:** This part contains the conversation history, a task-specific rubric, and the response to be evaluated. The rubrics are the same as those used in the dataset paper.
- **Assistant message:** This consists solely of the class label corresponding to the example (**Yes / No / To some extent**).

We experimented with two different training approaches:

- **Dimension-specific approach:** This involves training four separate models, each dedicated to one of the four evaluation dimensions (mistake identification, mistake location, providing guidance, and actionability). Each model is trained only on examples corresponding to its specific dimension.
- **Multi-dimension approach:** This involves training a single model using the combined training data from all four dimensions. The model is expected to infer the appropriate evaluation criteria based on the scoring rubric included in the user message.

The multi-dimension approach may help mitigate the class imbalance present in certain dimensions (particularly mistake identification), as the model is exposed to a more balanced distribution of the three class labels across different contexts.

The model was trained for three epochs with a batch size of 8 and a learning rate set to 2×10^{-4}, using a linear scheduler with a warm-up ratio of 0.03 and weight decay of 0.001. The training objective was next-token prediction, the same as in pre-training.

To train these models, we used the ClusterUY infrastructure (Nesmachnow and Iturriaga, 2019) with limited (and usually interrupted) access to NVIDIA A100 and NVIDIA A40 GPUs.

3.4.2 Inference

Once the models are fine-tuned, we perform inference using greedy decoding. The input prompt includes the system prompt and the user message, and the model is tasked with generating the assistant message. Since these are classification tasks, we perform a single forward pass and select the class label whose first token receives the highest logit value. Only the three candidate tokens (corresponding to the possible class labels) are considered, and the rest are ignored. This approach prevents hallucinations by constraining the model to produce one of the predefined labels.

We observed that with the previous method, the **To some extent** label was often under-predicted in favor of the **Yes** or **No** labels. To address this, we introduced an alternative method using thresholds defined separately for each dimension. We retrained the multi-dimension model (i.e. a single model for the first four tracks) on a subset of the training data and used the remaining examples as a validation set to tune the thresholds. The training/validation split was 80/20.

Using the fine-tuned model's predictions on the validation set, we computed the average probability of each class, grouped by the predicted label. Based on these statistics, we manually defined threshold rules using only the predicted probabilities for the **Yes** and **No** labels. Table 4 in Appendix B shows the threshold values we chose.

3.5 Results obtained over the dev set

We evaluated all these models on the `dev` set and the results are shown in Table 1.

The first observation we want to do is that even when classical ML algorithms did not manage to be the best in any track, they are still competitive. Some of them even achieved good performances, sometimes getting closer to the best model in the track. Secondly, we want to highlight that some sentence embeddings approaches performed better than using the fine-tuned Llama3.1 8B that we considered in the preliminary experiments. Finally, as expected, neural models performed the best.

Another interesting observation is that the `Llama 3.1 8B` LoRA fine-tuning, a model 16 times larger than Qwen and BERT, did not achieve significantly better results. In some dimensions, such as providing guidance and actionability, it even performed worse than the fine-tuned Qwen.

Overall, the best models in each track were as follows:

- Track 1 (Mistake Identification) – Sentence Embeddings and k-NN, using the balanced dataset
- Track 2 (Mistake Location) – Fine-tuning DistilBERT with thresholds
- Track 3 (Providing Guidance) – Fine-tuning Qwen using the multi-dimension approach and thresholds

Approach	Track 1		Track 2		Track 3		Track 4		Track 5	
	Mistake identification		Mistake location		Guidance		Actionability		Tutor identity	
	F1-macro	Accuracy	F1-macro	Accuracy	F1-macro	Accuracy	F1-macro	Accuracy	F1-macro	Accuracy
Preliminary experiments										
Always Yes	30.26	83.13	26.42	65.66	22.98	52.61	22.16	49.80	–	–
Random	23.95	33.73	26.94	31.93	30.02	31.33	28.99	30.72	11.75	12.05
Gemini Flash 2.0 Lite	50.74	76.71	48.49	62.25	46.54	55.62	–	–	–	–
Llama 3.1 8B (LoRA)	74.41	**92.37**	50.68	76.71	51.83	**63.25**	55.90	72.09	–	–
Classical Machine Learning										
RandomForest + TF-IDF (1-5)grams	71.46	90.76	47.24	74.10	44.08	60.64	52.00	64.26	70.25	69.68
RandomForest + TF-IDF (1-7)grams	60.14	87.55	40.17	60.64	40.93	52.21	42.00	58.23	69.66	68.47
RandomForest + TF-IDF (1-8)grams	71.04	90.56	46.92	74.30	44.37	60.44	50.19	62.25	68.13	66.87
SVC + TF-IDF (1-2)grams	71.82	91.37	49.26	75.50	43.10	61.04	48.42	65.06	79.16	77.71
SVC + TF-IDF (2-5)grams	60.47	88.96	43.74	73.96	40.68	60.64	40.92	60.04	74.76	73.69
k-NN ($k = 7$) + TF-IDF (1-2)grams	73.24	91.16	46.82	71.49	49.52	60.64	50.91	59.84	60.11	59.24
Fine-tuning DistilBERT										
DistilBERT	58.37	90.76	50.30	**76.91**	42.24	61.24	57.01	68.47	86.51	85.94
DistilBERT (thresholds)	63.04	88.35	**56.30**	67.47	52.22	55.22	53.02	66.67	–	–
BERT Embeddings + XGBoost	–	–	–	–	–	–	–	–	**87.74**	**87.14**
Sentence Embeddings										
e5 (response) k-NN ($k = 9$)	74.93	91.77	48.06	76.69	47.40	58.84	48.55	58.84	–	–
e5 (resp+hist) MLP	72.82	90.96	54.17	75.30	52.51	60.44	56.42	65.06	–	–
e5 (resp+hist+llm) MLP	73.54	91.16	54.36	75.30	52.10	59.84	56.51	65.46	–	–
e5 (resp) k-NN (balanced)	**79.16**	**92.37**	47.82	71.08	52.85	56.83	49.08	50.00	–	–
Fine-tuning Qwen										
Qwen (dimension-specific)	74.73	92.17	48.30	74.70	52.71	63.05	**61.20**	**73.29**	–	–
Qwen (multi-dimension)	73.04	91.37	51.96	74.50	53.26	61.45	59.57	68.47	–	–
Qwen (thresholds)	65.58	81.33	54.92	64.06	**54.18**	55.82	55.82	58.84	–	–

Table 1: Results for the five tracks over our dev set. In bold, the best results according to each metric for each track.

- Track 4 (Actionability) – Fine-tuning Qwen using the multi-dimension apporach
- Track 5 (Tutor identification) – BERT embeddings and XGBoost

4 Final submissions and experimental analysis

After evaluating all the previously described models on our dev set, we chose those which had the best performance, trying to ensure that at least one model of each category (*Classical machine learning*, *Fine-tuning DistilBERT*, *SentenceEmbeddings* and *Fine-tuning Qwen*) was used to predict the test instances in most of the tracks. The classical ML models were trained from scratch using both our `train` and dev set, while the neural models were only fine-tuned using the `train` set.

Table 2 shows the performance of our models in each track, the results we obtained and the resulting ranking position (#). To better understand the performance of our systems, we also considered quartiles for each track, and they are included in the table under the "Q" column. Taking a first look at the quartiles, we can see that none of our models was competitive enough to climb the rankings and finish in the first quartile. However, we want to highlight that in three out of the five tracks (Track1, Track3 and Track4) our models managed to finish in Q_2.

Overall, as when evaluating on the dev set, this time the neural models again achieved the best performance among our models. Moreover, something interesting to observe is that the fine-tuned Qwen

Submission	F1-macro	Accuracy	#	Q
Track 1 - Mistake identification				
e5 (resp.) k-NN (balanced)	**65.35**	84.49	56/153	Q_2
Qwen (dimension-specific)	64.94	86.68	62/153	Q_2
DistilBERT (thresholds)	64.30	85.20	64/153	Q_2
SVC + TF-IDF	59.11	84.81	104/153	Q_3
e5 (response) k-NN ($k = 9$)	58.39	84.36	110/153	Q_3
Track 2 - Mistake location				
DistilBERT (thresholds)	**49.58**	58.63	47/86	Q_3
Qwen (multi-dimension)	49.52	70.78	49/86	Q_3
e5 (resp+hist) MLP	49.40	67.36	51/86	Q_3
Qwen (thresholds)	49.13	55.20	54/86	Q_3
SVC + TF-IDF	45.85	70.39	72/86	Q_4
Track 3 - Providing guidance				
Qwen (multi-dimension)	**50.49**	59.47	36/105	Q_2
DistilBERT (thresholds)	49.19	53.85	48/105	Q_2
Qwen (thresholds)	47.53	50.36	64/105	Q_3
k-NN + TF-IDF	47.41	59.21	66/105	Q_3
e5 (resp+hist) MLP	47.14	57.85	71/105	Q_3
Track 4 - Actionability				
Qwen (dimension-specific)	**61.28**	70.33	42/87	Q_2
Qwen (multi-dimension)	60.54	68.00	46/87	Q_3
e5 (resp+hist) MLP	56.37	63.22	60/87	Q_3
DistilBERT (thresholds)	52.61	64.12	68/87	Q_4
RandomForest + TF-IDF	51.91	62.64	70/87	Q_4
Track 5 - Tutor identification				
BERT + XGBoost	**83.85**	84.74	27/54	Q_3
DistilBERT	**83.85**	84.74	28/54	Q_3
SVC + TF-IDF	80.44	80.22	39/54	Q_3
BERT + Educated guess	68.16	68.65	42/54	Q_4

Table 2: Results for the five tracks over the competition's test data. The "#" column indicates the position the system got in the rankings, and the "Q" column indicates the quartile related to that position (splitting in 4 buckets the number of participants in each track).

Track	**Rank** / Total	**Q**	**Δ Exact F_1**	**Δ Exact Accuracy**	**Δ Lenient F_1**	**Δ Lenient Accuracy**
Track 1	23 / 44	Q_3	$71.81 - 65.35 = 06.46$	$86.23 - 84.49 = 01.74$	$89.57 - 83.95 = 05.62$	$94.57 - 91.92 = 02.65$
Track 2	21 / 32	Q_3	$59.83 - 49.59 = 10.24$	$76.79 - 58.63 = 18.16$	$83.86 - 72.00 = 11.86$	$86.30 - 76.08 = 10.22$
Track 3	17 / 35	Q_3	$58.34 - 50.49 = 07.85$	$66.13 - 59.47 = 06.66$	$77.98 - 70.57 = 07.41$	$81.90 - 77.51 = 04.39$
Track 4	17 / 29	Q_3	$70.85 - 61.29 = 09.56$	$72.98 - 70.33 = 02.65$	$85.27 - 82.72 = 02.55$	$88.37 - 85.59 = 02.78$
Track 5	12 / 20	Q_3	$96.98 - 83.85 = 13.13$	$96.64 - 84.75 = 11.89$	*N/A*	*N/A*

Table 3: Performance difference between our best submissions and the winners, for each task. This table was built based on the *team results*, so the total number of submissions for each track is always fewer than those considered in Table 2.

models and the BERT models got similar performance. This seems to indicate that the generative capabilities of Qwen are good enough to also work as an emergent classifier.

Most of the models we used are not well-suited for handling long contexts. This led us to question how essential the *conversation history* truly is for assessing the four evaluation dimensions, or whether the model's response alone is enough to obtain good results. Therefore, we tested training some models both with and without including the conversation history as input. In this regard, the most significant experiments were those using sentence embeddings. These experiments show that nearly every dimension benefits from the inclusion of history, except for the mistake identification dimension, which performs notably better without it (i.e. using only the tutor's response). More broadly, Qwen-based methods (which incorporate the full conversation history) achieve the best results in providing guidance and actionability, and, in contrast, BERT-based methods (which do not use the conversation history) perform better on mistake location. This pattern suggests that more subtle dimensions like guidance and actionability benefit more from access to the full conversational context. Further experimentation is required to validate all these preliminary observations.

Finally, while the DistilBERT with XGBoost approach seemed to have a good performance on our dev set, its final performance (on the test set) was identical to that of the DistilBERT fine-tuning model (without XGBoost). This was not the only difference we had between our dev set and the `test` set. As can be seen by comparing Tables 1 and 2, most methods performed noticeably better on our internal dev set than on the `test` set. We believe this performance gap may be due to differences in the class distributions between the two sets.

Furthermore, the experiment using undersampling to balance the classes showed a significant improvement on the `test` set, going from being the worst-performing submission to being the best one. This further highlights the impact of class imbalance on model performance.

4.1 How far were these lightweight models from winning?

Finally, to answer our *research question*, we wanted to check how far our lightweight models went in the shared task. Beyond the ranking positions, we wanted to focus on *how big* (according to the official scores) was the gap between these models and those that settled the state of the art, winning the competition. In Table 3 we show the difference (Δ) — for each metric — of our best predictions with the winner team in each track[6]. As a reference, we also include our team's position in that track and the correspondent *quartile* (this time, based on the number of teams, and not on the number of submitted systems).

Taking a look at the table, we can see that, according to Δ Exact F_1, the closest gap between our performance and the winning team was 06.46 (in Track 1), while the biggest gap was 13.13 (in Track 5). We think this difference in performance is very small considering the restrictions we had.

5 Conclusions

In this paper we presented the RETUYT-INCO participation at the 2025 BEA shared task, characterized by our self-imposed restriction of only using models under 1B parameters. Although our research lab have access to cheap API LLMs and very limited access to run 7B LLMs on clusters, we are conscious that this is not the case for other research labs in the Global South, that usually work

[6]Since the organizers considered the *Exact* F_1 metric as the main one, we considered as *winning teams* those which got the highest score according to that metric. Therefore, for all metrics, we calculated the Δ according to the score achieved by the winning team in that track. This way, even if a different team got a better result according to other metric, we still calculated the Δ according to the winning team.

in even deeper under-resourced scenarios. Our self-imposed restriction tries to represent this scenario.

Overall, we used classical machine learning models, BERT-based models, and a QWEN 0.5B LLM. Despite their (very small) size we finished in mid-ranking positions. Beyond the results in the rankings, the result we want to highlight is that the gaps in performance we had with the winning teams were between 6.46 and 13.13 F_1 *exact* points.

We find that gap surprisingly small, taking into account that we did not use LLMs bigger than 1B, nor paid for API access, nor paid for premium cloud computing, nor needed top-tier resources to run our experiments. Additionally, following the environmental concerns that surround the carbon footprint of state-of-the-art LLMs (Luccioni et al., 2023; Faiz et al., 2024; Liu and Yin, 2024), we consider this an interesting tradeoff: to sacrifice some performance, in order to have models that do not need extensive training/inference time or power, but that are still competent. Based on all of the above, we think research on models that run on low-cost GPUs — or need no GPU at all — should definitely go on.

6 Limitations

Throughout the paper we have outlined several limitations we have to run experiments with large models. These constraints led to our self-imposed restriction of using only neural models with fewer than 1B parameters. Naturally, our work does not present state-of-the-art results, nor does it intend to. Furthermore, we prioritized breadth (i.e. trying many model types) over depth (i.e. optimizing a single approach or architecture extensively). While this gives a broader perspective on the diverse possibilities that lightweight models have to offer, it may have limited the performance ceiling of individual models.

Regarding our methodology, we made the decision of splitting the full set into two subsets (`train` and `dev`) considering as a priority to keep the conversations and their responses in the same subset. This decision may have introduced some noise and class imbalance, since we found remarkable differences in the performance of our models over the `dev` set and the final `test` set (after submission). Since the fine-tuned models and the thresholds used were adjusted specifically to our `dev` set, they may not generalize well to other similar corpora.

Finally, and related to the previous considerations, we did not systematically perform hyperparameter tuning due to both hardware and time limitations. Additionally, prior to our final submissions, we only trained (from scratch) the classical ML models on the full set (our `train` + `dev` sets). Since the neural models were our best approaches, searching better hyperparameters and training them with more data could have made the performance gaps a bit smaller.

Acknowledgments

This paper has been funded by ANII (Uruguayan Innovation and Research National Agency), Grant No. $FMV\_1\_2023\_1\_176581$.

References

AI@Meta. 2024. Llama 3 model card.

Alexis Baladón, Ignacio Sastre, Luis Chiruzzo, and Aiala Rosá. 2023. RETUYT-InCo at BEA 2023 shared task: Tuning open-source LLMs for generating teacher responses. In *Proceedings of the 18th Workshop on Innovative Use of NLP for Building Educational Applications (BEA 2023)*, pages 756–765, Toronto, Canada. Association for Computational Linguistics.

Tianqi Chen and Carlos Guestrin. 2016. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794.

Luis Chiruzzo, Laura Musto, Santiago Gongora, Brian Carpenter, Juan Filevich, and Aiala Rosa. 2022. Using NLP to support English teaching in rural schools. In *Proceedings of the Second Workshop on NLP for Positive Impact (NLP4PI)*, pages 113–121, Abu Dhabi, United Arab Emirates (Hybrid). Association for Computational Linguistics.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. *CoRR*, abs/1911.02116.

Ahmad Faiz, Sotaro Kaneda, Ruhan Wang, Rita Chukwunyere Osi, Prateek Sharma, Fan Chen, and Lei Jiang. 2024. Llmcarbon: Modeling the end-to-end carbon footprint of large language models. In *The Twelfth International Conference on Learning Representations*.

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Oana Ignat, Zhijing Jin, Artem Abzaliev, Laura Biester, Santiago Castro, Naihao Deng, Xinyi Gao, Aylin Ece Gunal, Jacky He, Ashkan Kazemi, Muhammad Khalifa, Namho Koh, Andrew Lee, Siyang Liu, Do June Min, Shinka Mori, Joan C. Nwatu, Veronica Perez-Rosas, Siqi Shen, Zekun Wang, Winston Wu, and Rada Mihalcea. 2024. Has it all been solved? open NLP research questions not solved by large language models. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 8050–8094, Torino, Italia. ELRA and ICCL.

Ekaterina Kochmar, Kaushal Kumar Maurya, Kseniia Petukhova, K. V. Aditya Srivatsa, Anaïs Tack, and Justin Vasselli. 2025. Findings of the BEA 2025 Shared Task on Pedagogical Ability Assessment of AI-powered Tutors. In *Proceedings of the 20th Workshop on Innovative Use of NLP for Building Educational Applications*.

Vivian Liu and Yiqiao Yin. 2024. Green ai: exploring carbon footprints, mitigation strategies, and trade offs in large language model training. *Discover Artificial Intelligence*, 4(1):49.

Alexandra Sasha Luccioni, Sylvain Viguier, and Anne-Laure Ligozat. 2023. Estimating the carbon footprint of bloom, a 176b parameter language model. *Journal of Machine Learning Research*, 24(253):1–15.

Kaushal Kumar Maurya, Kv Aditya Srivatsa, Kseniia Petukhova, and Ekaterina Kochmar. 2025. Unifying AI tutor evaluation: An evaluation taxonomy for pedagogical ability assessment of LLM-powered AI tutors. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 1234–1251, Albuquerque, New Mexico. Association for Computational Linguistics.

Sergio Nesmachnow and Santiago Iturriaga. 2019. Cluster-uy: Collaborative scientific high performance computing in uruguay. In *Supercomputing*, pages 188–202, Cham. Springer International Publishing.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Aiala Rosá, Santiago Góngora, Juan Pablo Filevich, Ignacio Sastre, Laura Musto, Brian Carpenter, and Luis Chiruzzo. 2025. A platform for generating educational activities to teach english as a second language. *Preprint*, arXiv:2504.20251.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2020. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *Preprint*, arXiv:1910.01108.

Ignacio Sastre, Leandro Alfonso, Facundo Fleitas, Federico Gil, Andrés Lucas, Tomás Spoturno, Santiago Góngora, Aiala Rosá, and Luis Chiruzzo. 2024. RETUYT-INCO at MLSP 2024: Experiments on language simplification using embeddings, classifiers and large language models. In *Proceedings of the 19th Workshop on Innovative Use of NLP for Building Educational Applications (BEA 2024)*, pages 618–626, Mexico City, Mexico. Association for Computational Linguistics.

Matthew Shardlow, Fernando Alva-Manchego, Riza Batista-Navarro, Stefan Bott, Saul Calderon Ramirez, Rémi Cardon, Thomas François, Akio Hayakawa, Andrea Horbach, Anna Hülsing, Yusuke Ide, Joseph Marvin Imperial, Adam Nohejl, Kai North, Laura Occhipinti, Nelson Peréz Rojas, Nishat Raihan, Tharindu Ranasinghe, Martin Solis Salazar, Sanja Štajner, Marcos Zampieri, and Horacio Saggion. 2024. The BEA 2024 shared task on the multilingual lexical simplification pipeline. In *Proceedings of the 19th Workshop on Innovative Use of NLP for Building Educational Applications (BEA 2024)*, pages 571–589, Mexico City, Mexico. Association for Computational Linguistics.

Anaïs Tack, Ekaterina Kochmar, Zheng Yuan, Serge Bibauw, and Chris Piech. 2023. The BEA 2023 shared task on generating AI teacher responses in educational dialogues. In *Proceedings of the 18th Workshop on Innovative Use of NLP for Building Educational Applications (BEA 2023)*, pages 785–795, Toronto, Canada. Association for Computational Linguistics.

Qwen Team. 2024. Qwen2.5: A party of foundation models.

Liang Wang, Nan Yang, Xiaolong Huang, Linjun Yang, Rangan Majumder, and Furu Wei. 2024. Multilingual e5 text embeddings: A technical report. *arXiv preprint arXiv:2402.05672*.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, Jin Xu, Jingren Zhou, Jinze Bai, Jinzheng He, Junyang Lin, Kai Dang, Keming Lu, Keqin Chen, Kexin Yang, Mei Li, Mingfeng Xue, Na Ni, Pei Zhang, Peng Wang, Ru Peng, Rui Men, Ruize Gao, Runji Lin, Shijie Wang, Shuai Bai, Sinan Tan, Tianhang Zhu, Tianhao Li, Tianyu Liu, Wenbin Ge, Xiaodong Deng, Xiaohuan Zhou, Xingzhang Ren, Xinyu Zhang, Xipin Wei, Xuancheng Ren, Yang Fan, Yang Yao, Yichang Zhang, Yu Wan, Yunfei Chu, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zhihao Fan. 2024. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

A Prompts

This Appendix presents the prompts used for fine-tuning the decoder-only language models, as explained in Section 3.4.

System prompt

```
You are a critic evaluating a tutor's
interaction with a student, responsible
for providing a clear and objective single
evaluation based on specific criteria.
Each assessment must accurately reflect the
absolute performance standards.
```

User message

```
# Previous Conversation between Tutor and
Student:
{history}

# Scoring Rubric:
{rubric}

# Tutor Response:
{response}
```

Mistake identification rubric

```
[Has the tutor identified a mistake in a
student's response?]
1) Yes
2) To some extent
3) No
```

Mistake location rubric

```
[Does the tutor's response accurately point
to a genuine mistake and its location?]
1) Yes
2) To some extent
3) No
```

Providing guidance rubric

```
[Does the tutor offer correct and relevant
guidance, such as an explanation, ela-
boration, hint, examples, and so on?]
1) Yes (guidance is correct and relevant
to the mistake)
2) To some extent (guidance is provided but
it is fully or partially incorrect or
incomplete)
3) No
```

Actionability rubric

```
[Is it clear from the tutor’s feedback what
the student should do next?]
1) Yes
2) To some extent
3) No
```

B Qwen Thresholds

Table 4 presents the thresholds used with the Qwen model, as explained in Section 3.4.

Dimension	**Yes condition**	**No condition**	**TSE condition**
Mistake Identification	Yes > 0.90 & No < 0.05	Yes < 0.40 & No > 0.50	Otherwise
Mistake Location	Yes > 0.75 & No < 0.15	Yes < 0.42 & No > 0.50	Otherwise
Providing Guidance	Yes > 0.65 & No < 0.12	Yes < 0.35 & No > 0.45	Otherwise
Actionability	Yes > 0.70 & No < 0.14	Yes < 0.25 & No > 0.65	Otherwise

Table 4: Threshold-based classification rules for each evaluation dimension using Qwen. TSE = "To some extent".