Large Language Models in the Task of Automatic Validation of Text Classifier Predictions

Aleksandr Tsymbalov
Independent Researcher
9060860094s@gmail.com

Abstract

Machine learning models for text classification are trained to predict a class for a given text. To do this, training and validation samples must be prepared: a set of texts is collected, and each text is assigned a class. These classes are usually assigned by human annotators with different expertise levels, depending on the specific classification task. Collecting such samples from scratch is labor-intensive because it requires finding specialists and compensating them for their work; moreover, the number of available specialists is limited, and their productivity is constrained by human factors. While it may not be too resource-intensive to collect samples once, the ongoing need to retrain models (especially in incremental learning pipelines He et al. (2020)) to address data drift (also called model drift IBM (2024)) makes the data collection process crucial and costly over the model's entire lifecycle. This paper proposes several approaches to replace human annotators with Large Language Models (LLMs) to test classifier predictions for correctness, helping ensure model quality and support high-quality incremental learning.

1 Introduction

The creation of reliable text classifiers typically demands a carefully annotated gold-standard corpus, benchmark metrics, and continuous monitoring for data drift - tasks that are costly because they rely on domain-specialist human annotators. With the advent of large language models (LLMs), which already demonstrate broad knowledge and flexible reasoning Sahoo et al. (2025); Vatsal & Dubey (2024), researchers are investigating whether these models can augment or even replace human annotators. Simply adding a second conventional classifier would double the maintenance effort without reducing the annotation needs. In the case examined, detecting Russian-language client intents in support chat messages, misclassification prevents automated client request closure and forces human intervention. Hence, high-quality annotation remains critical for maintaining classifier performance. This study therefore evaluates the feasibility of substituting human annotation with LLM-based annotation while preserving or improving the overall quality of the classifier.

2 Social impacts statement

Large-scale **LLM-based automation of data annotation** can transform supervised NLP, cutting costs and latency, but it must be deployed responsibly.

Key benefits. (i) *Democratisation*: automating access to high-quality data for smaller firms, research groups, and low-resource languages. (ii) *Safety by design*: a single-token interface plus calibrated *"unk"* (model rejection of annotation or classification) thresholds defer low-confidence cases to humans, preserving a transparent human-in-the-loop workflow. (iii) *Quality uplift*: across benchmarks the ensemble beats regular annotators on accuracy–coverage and matches expert gold labels, raising the production "floor" for classifier quality. Section F describes the types of human annotators.

Risks & mitigations. *Labour displacement*: reduce routine work but reallocate annotators to edge-case triage and fairness audits, and provide reskilling paths. *Bias amplification*: periodic bias audits on stratified test suites and diverse-model ensembles curb shared blind spots. *Privacy*: keep data on-premises, log access, and add differential privacy noise when sharing artefacts. *Environmental cost*: GPU inference is energy-intensive; model distillation, quantisation, and renewable hosting mitigate net carbon versus distributed crowdsourcing. *Long-term accountability*: dynamic threshold learning keeps abstention rates aligned with data drift, and lightweight explainability interfaces expose rationale snippets without leaking sensitive text.

3 Related work

Research on leveraging LLMs for classification falls into two streams: **(i) LLMs as direct classifiers via in-context learning (ICL) Brown et al. (2020)** and **(ii) LLMs as data annotators**.

3.1 LLMs as ICL classifiers

Recent work shows that a few carefully chosen demonstrations let GPT-4 outperform fine-tuned masked-LMs on the 77-class *banking77* benchmark Loukas et al. (2023); Casanueva et al. (2020); OpenAI (2023), albeit at non-trivial inference cost. Follow-up studies report that GPT-4 excels on "generalization-heavy" tasks, whereas smaller fine-tuned models match or beat it on easier ones; open-source Llama-2 can occasionally beat GPT-3.5 Yu et al. (2023); Touvron et al. (2023); Ye et al. (2023). Multi-stage prompting (e.g., reranking with varied prompts) further boosts GPT-4o on query–document matching Schnabel et al. (2025); OpenAI (2024).

3.2 LLMs as data annotators

Another line replaces or assists humans during labeling. Zero-shot GPT-3.5 produced higher-quality annotations than humans in 4/5 task types at 20× lower cost Gilardi et al. (2023). Distillation of LLM-generated labels can train competitive classifiers quickly and cheaply Pangakis & Wolken (2024). LLM annotation even works for medical images, yielding strong CNNs from GPT-labeled X-rays Al Mohamad et al. (2025). Selective use of GPT-3.5 when "confident" cuts labeling effort, though self-estimated confidence can be unreliable Rouzegar & Makrehchi (2024); Li et al. (2024).

Limitations of prior work. 1) Many public datasets Loukas et al. (2023); Schnabel et al. (2025); Rouzegar & Makrehchi (2024) were likely seen during LLM pre-training; our dataset (Section 4.1) is novel. **2)** Tasks usually have few classes; ours (Section 4.1.1) has >200. **3)** Most studies rely on proprietary models; we focus on open-weight LLMs (Section I). **4)** Text-based label extraction blurs rejection zones; we instead return calibrated probabilities (Section 4.3.2).

4 Methodology

4.1 Dataset

4.1.1 Intent

A client intent is the customer's underlying question or requested action. The taxonomy contains ≈ 250 intents, some of which are closely related, so multi-stage retraining is used to push borderline classes farther apart. Every intent has a definition plus typical and contentious examples. Two meta-labels complete the set: **unk** ("cannot classify") and **conf** ("multiple plausible intents", used only for analysis; see Section 4.2.2).

4.2 Human-grounded annotation

One common use of human annotation is to verify the correctness of predictions made by a text classification model that outputs a large number of classes (a classic text classification scenario). To maintain annotation quality, the same instance is shown to multiple annotators, and an annotation is deemed correct if their answers coincide.
Assume that the classification model can return K best probability predictions, then two types of annotations can be used to evaluate the model's performance.

4.2.1 Binary annotation

Let C represent the predicted class with the highest probability of the text classifier, T is client text. Then:

$$\mathbf{F}\colon \mathbf{T} \times \mathbf{C} \to \{\text{“0”}, \text{“1”}\}, \quad \text{where } |\mathbf{C}| \approx 250,\ |\mathbf{T}| = \infty,$$

where F is an annotator who returns a response of 0 (*”no”*, *”the class does not match the text, classification model made an incorrect prediction”*) and a response of 1 (*”yes”*, *”the class matches the text, classification model made a correct prediction”*).

4.2.2 Multi-class annotation

Let $C = \{c_1, c_2, c_3, ..., c_n\}$ be the set of all possible classes, and T be a set of texts.
Then:

$$\mathbf{F}\colon \mathbf{T} \times \mathbf{C}^5 \to \mathbf{C}, \quad \text{where } |\mathbf{C}| \approx 250,\ |\mathbf{T}| = \infty,$$

$$\mathbf{F}\big(t, (c_{i_1}, \ldots, c_{i_5})\big) \ \in\ \{c_{i_1}, \ldots, c_{i_5}\}$$

$$\forall\, t \in \mathbf{T},\ (c_{i_1}, \ldots, c_{i_5}) \in \mathbf{C}^5,$$

where C^5 contains the classifier's top-5 candidates for t.

Annotators pick the best label from these five, resolving tight semantic ties. As the model is retrained on their choices, cross-entropy naturally widens margins between close intents, and prior internal tests confirm that a 1–5 shortlist almost always covers the true class while easing annotator load.

A special flag, *”conf”* (”multiple intentions”), may be used for analysis when a request genuinely blends intents, but in regular annotation annotators must still choose one of the supplied labels - *”unk”* (refusal) is always among them.

4.3 LLM annotation

This paper proposes two ways to obtain the LLM annotation. Depending on your requirements, you can choose one or the another. Combining approaches is not effective; the reasons are described in Section D.2.

4.3.1 Text-approach

The ”text-approach” extracts the predicted label from the LLM's generated text. Because it treats the model as a black box - no probability access needed - most studies adopt it. The natural-language rationale it returns supports prompt tinkering, multi-stage setups (a second LLM audits the first), and selective tool calls (extra analysis only when uncertainty is high). Its downside is a fuzzy rejection boundary, so for multi-class tasks we let the ”refuse” label (*”unk”*) compete for a spot in the top-5 predictions, noting that the LLM may already output a refusal on its own.

4.3.2 Prob-approach

The ”prob-approach” (Section E, Fig. 4) queries the likelihood of the first generated token. The prompt forces the LLM to answer with ”0/1” for binary or a class index for multi-class

tasks, giving true label probabilities. A rejection zone (*"unk"*) is introduced by thresholding: if the highest class probability among the N options falls below the cutoff, the LLM refuses and the item is routed to a human annotator; otherwise the most-probable label is returned. Because only one token is produced, no reasoning is available - Section D.2 shows that allowing free-form reasoning actually degrades annotation metrics compared with this single-token scheme.

4.4 Metrics

Tables with all metrics are provided in Section A.

4.5 Proposed approach

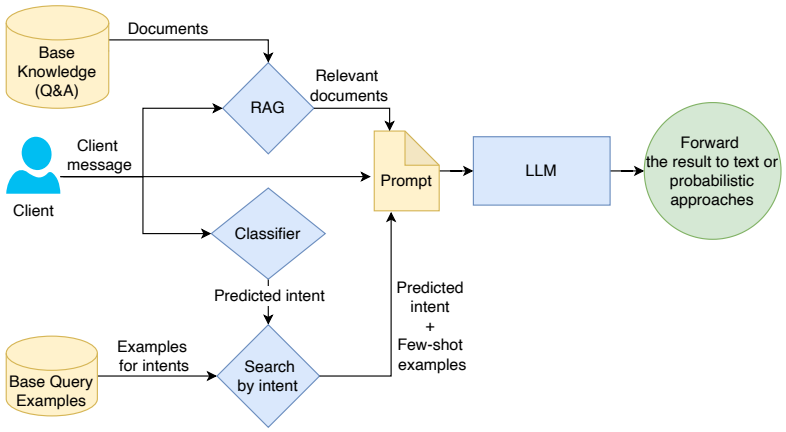

Figure 1: Proposed pipeline

The final proposed approach is shown in Fig. 1: First, the client's request is sent to the classifier and RAG. The RAG system searches the knowledge base for relevant documents, which are then sent to the prompt. The classifier model predicts a class to the client request, then sends one or more predicted intents to a dictionary where the key is the intent and the values are examples of the intent and description. The values are then forwarded to the prompt. Description of prompts in Section G. Prompts are presented in Section P.

Ensembles can be used to modify this approach. This requires obtaining token probabilities using a prob-approach for multiple LLMs.

5 Results

Best RAG: Table N.3, Table N.5. Best LoRA: Table K.3. Metrics: Section N, Section K

5.1 Multi-class

Table 1 shows the best approach, which consists of averaging the prediction probabilities of two LLMs and applying a tuned threshold. By slightly reducing the coverage with threshold, it is possible to achieve an increase in the accuracy of the LLM separately and of the ensemble as a whole. We achieved approximately the same accuracy on intents as human annotators and the highest coverage.

MODEL	COVERAGE ("conf"="unk")	ACCURACY (w/o "conf")	$\text{ACCURACY}_{\text{total}}$
QWEN2.5-32B BEST-RAG PROB-APPROACH USE THRESHOLDS	**0.4376**	0.6831	0.2989
LLAMA3.3-70B BEST-RAG PROB-APPROACH USE THRESHOLDS	0.4374	0.6758	0.2957
HUMAN	0.2134	0.6977	0.1489
ENSEMBLE LLAMA3.3-70B BEST-RAG & QWEN2.5-32B BEST-RAG PROB-APPROACH USE THRESHOLDS	0.4269	**0.7030**	**0.3001**

Table 1: Metrics for ensemble (Multi-class, Prob-approach)

MODEL	ACC.	F1-SCORE (MACRO. AVG.)	COVERAGE (PERC.)
QWEN2.5-32B BEST-RAG BEST-LORA	0.8430	**0.8405**	63%
LLAMA3.3-70B BEST-RAG	0.8113	0.8106	61%
HUMAN	0.8018	0.7977	100%
HUMAN BASED ON THE SAME 65% DATA AS LLM	0.8174	0.8133	65%
ENSEMBLE LLAMA3.3-70B BEST-RAG & QWEN2.5-32B BEST-RAG BEST-LORA	**0.8851**	0.8292	**65%**

Table 2: Metrics for ensemble (Binary, Prob-approach)

5.2 Binary

In the case of binary annotation (Table 2), many of the approaches described in the paper outperformed human metrics, therefore the goal was to increase coverage without significantly lowering metrics. The quality of annotation on the same data (65%) as humans is higher for LLMs. This may mean that LLM annotates simple tasks better than humans, but with more complex tasks (where the probability of answers is below the threshold), human annotators still perform better.
Fig. 7 (Section M) demonstrates that the increase in accuracy with a decrease in the threshold (decrease in coverage) is non-linear.

6 Conclusion

We showed that Large Language Models (LLMs) can **replace - or substantially supplement - human annotators** in a large-scale, multi-intent text-classification pipeline without degrading quality.

Two paradigms. (i) *Text-approach:* the LLM produces a natural-language rationale and label. (ii) *Prob-approach:* the LLM outputs a single token; its calibrated probability is thresholded to accept or defer.

Enhancements. LoRA fine-tuning (reasoning & classifier heads), retrieval-augmented generation (RAG) for domain documents, and heterogeneous LLM *ensembles* further boosted coverage and confidence (Secs. K.1, D.3). Error propagation to the downstream classifier is quantified in App. M.

Results. On both binary and multi-class benchmarks our best pipelines *matched or exceeded* regular human annotators (Tabs. 1– K.3) while cutting average annotation time $\times 8$ (Sec. H).

Implication. LLMs can act as primary annotators in production, enabling continuous, high-quality incremental learning at scale.

Future work. We will study adaptive thresholds, transfer to new intent taxonomies, and richer human-LLM collaboration interfaces.

References

Fares Al Mohamad, Leonhard Donle, Felix Dorfner, Laura Romanescu, Kristin Drechsler, Mike P. Wattjes, Jawed Nawabi, Marcus R. Makowski, Hartmut Häntze, Lisa Adams, Lina Xu, Felix Busch, Aymen Meddeb, and Keno Kyrill Bressem. Open-source large language models can generate labels from radiology reports for training convolutional neural networks. *Academic Radiology*, 32(5):2402–2410, 2025. doi: 10.1016/j.acra.2024.12.028. URL https://www.sciencedirect.com/science/article/pii/S1076633224009966.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are Few-Shot learners. 2020.

Iñigo Casanueva, Tadas Temčinas, Daniela Gerz, Matthew Henderson, and Ivan Vulić. Efficient intent detection with dual sentence encoders, 2020. URL https://arxiv.org/abs/2003.04807.

Jianlv Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. BGE M3-embedding: Multilingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation, 2024. URL https://arxiv.org/abs/2402.03216.

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jiawei Wang, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, R. L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, S. S. Li, Shuang Zhou, Shaoqing Wu, Shengfeng Ye, Tao Yun, Tian Pei, Tianyu Sun, T. Wang, Wangding Zeng, Wanjia Zhao, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, W. L. Xiao, Wei An, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, X. Q. Li, Xiangyue Jin, Xiaojin Shen, Xiaosha Chen, Xiaowen Sun, Xiaoxiang Wang, Xinnan Song, Xinyi Zhou, Y. K. Wang, Y. Q. Wang, Y. X. Wei, Yang Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Yu, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yuan Ou, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yunfan Xiong, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Y. X. Zhu, Yanhong Xu, Yanping Huang, Yaohui Li, Yi Zheng, Yuchen Zhu, Yunxian Ma, Ying Tang, Yukun Zha, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu Zhang, and Zhen Zhang. Deepseek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning, 2025. URL https://arxiv.org/abs/2501.12948.

Matthijs Douze, Alexandr Guzhva, Chengqi Deng, Jeff Johnson, Gergely Szilvasy, Pierre-Emmanuel Mazaré, Maria Lomeli, Lucas Hosseini, and Hervé Jégou. The Faiss library. 2024. URL https://arxiv.org/abs/2401.08281.

Fabrizio Gilardi, Meysam Alizadeh, and Maël Kubli. Chatgpt outperforms crowd workers for text-annotation tasks. *Proceedings of the National Academy of Sciences*, 120(30), July 2023.

ISSN 1091-6490. doi: 10.1073/pnas.2305016120. URL http://dx.doi.org/10.1073/pnas.2305016120.

Jiangpeng He, Runyu Mao, Zeman Shao, and Fengqing Zhu. Incremental learning in online scenario, 2020. URL https://arxiv.org/abs/2003.13191.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models, 2021. URL https://arxiv.org/abs/2106.09685.

IBM. Model drift: Why it happens and how to detect it. https://www.ibm.com/think/topics/model-drift, 2024. Accessed: 2024-07-16.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with PagedAttention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*, 2023.

Moxin Li, Wenjie Wang, Fuli Feng, Fengbin Zhu, Qifan Wang, and Tat-Seng Chua. Think twice before trusting: Self-detection for large language models through comprehensive answer reflection, 2024. URL https://arxiv.org/abs/2403.09972.

Lefteris Loukas, Ilias Stogiannidis, Prodromos Malakasiotis, and Stavros Vassos. Breaking the bank with ChatGPT: Few-shot text classification for finance, 2023. URL https://arxiv.org/abs/2308.14634.

Nexusflow. Introducing athene-v2: Advancing beyond the limits of scaling with targeted post-training. https://nexusflow.ai/blogs/athene-v2, November 2024. Blog post announcing Athene-V2-Chat-72B.

Aleksandr Nikolich, Konstantin Korolev, Sergei Bratchikov, Igor Kiselev, and Artem Shelmanov. Vikhr: Constructing a state-of-the-art bilingual open-source instruction-following large language model for Russian. In *Proceedings of the 4th Workshop on Multilingual Representation Learning (MRL) @ EMNLP 2024*. Association for Computational Linguistics, 2024. URL https://arxiv.org/abs/2405.13929.

NVIDIA Corporation and Mistral AI. NeMo Mistral toolkit for large language models. https://developer.nvidia.com/nemo/mistral, 2024. Accessed: 2025-05-16.

OpenAI. GPT-4 technical report. *CoRR*, abs/2303.08774, 2023. doi: 10.48550/ARXIV.2303.08774. URL https://doi.org/10.48550/arXiv.2303.08774.

OpenAI. GPT-4o system card. https://openai.com/index/gpt-4o-system-card/, 2024. Accessed: 2025-05-16.

Nicholas Pangakis and Samuel Wolken. Knowledge distillation in automated annotation: Supervised text classification with LLM-generated training labels, 2024. URL https://arxiv.org/abs/2406.17633.

Qwen Team. Qwen2.5: A party of foundation models. https://qwenlm.github.io/blog/qwen2.5/, September 2024.

Stephen E. Robertson and Hugo Zaragoza. The probabilistic relevance framework: BM25 and beyond. *Foundations and Trends in Information Retrieval*, 3(4):333–389, 2009.

Hamidreza Rouzegar and Masoud Makrehchi. Enhancing text classification through LLM-driven active learning and human annotation, 2024. URL https://arxiv.org/abs/2406.12114.

Pranab Sahoo, Ayush Kumar Singh, Sriparna Saha, Vinija Jain, Samrat Mondal, and Aman Chadha. A systematic survey of prompt engineering in large language models: Techniques and applications, 2025. URL https://arxiv.org/abs/2402.07927.

Julian A. Schnabel, Johanne R. Trippas, Falk Scholer, and Danula Hettiachchi. Multi-stage large language model pipelines can outperform GPT-4o in relevance assessment, 2025. URL https://arxiv.org/abs/2501.14296.

Artem Snegirev, Maria Tikhonova, Anna Maksimova, Alena Fenogenova, and Alexander Abramov. The russian-focused embedders' exploration: ruMTEB benchmark and russian embedding model design, 2024. URL https://arxiv.org/abs/2408.12503.

Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Hussenot Léonard, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, Peter Liu, Pouya Tafti, Abe Friesen, Sabela Ramos, Ravin Kumar, Charline Le Lan, Sammy Jerome, Anton Tsitsulin, Nino Vieillard, Piotr Stanczyk, Sertan Girgin, Nikola Momchev, Matt Hoffman, Shantanu Thakoor, Jean-Bastien Grill, Behnam Neyshabur, Olivier Bachem, Alanna Walton, Aliaksei Severyn, Alicia Parrish, Aliya Ahmad, Allen Hutchison, Alvin Abdagic, Amanda Carl, Amy Shen, Andy Brock, Andy Coenen, Anthony Laforge, Antonia Paterson, Ben Bastian, Bilal Piot, Bo Wu, Brandon Royal, Charlie Chen, Chintu Kumar, Chris Perry, Chris Welty, Christopher A. Choquette-Choo, Danila Sinopalnikov, Francesco Visin, Gabriel Rasskin, Gary Wei, Glenn Cameron, Gus Martins, Hadi Hashemi, Hanna Klimczak-Plucińska, Harleen Batra, Harsh Dhand, Ivan Nardini, Jacinda Mein, Jack Zhou, James Svensson, Jeff Stanway, Jetha Chan, Jin Peng Zhou, Joana Carrasqueira, Joana Iljazi, Jocelyn Becker, Joe Fernandez, Joost van Amersfoort, Josh Gordon, Josh Lipschultz, Josh Newlan, Ju-yeong Ji, Kareem Mohamed, Kartikeya Badola, Kat Black, Katie Millican, Keelin McDonell, Kelvin Nguyen, Kiranbir Sodhia, Kish Greene, Lars Lowe Sjoesund, Lauren Usui, Laurent Sifre, Lena Heuermann, Leticia Lago, Lilly McNealus, Livio Baldini Soares, Logan Kilpatrick, Lucas Dixon, Luciano Martins, Machel Reid, Manvinder Singh, Martin Görner, Mat Velloso, Mateo Wirth, Matt Davidow, Matt Miller, Matthew Rahtz, Matthew Watson, Meg Risdal, Mehran Kazemi, Michael Moynihan, Ming Zhang, Minsuk Kahng, Minwoo Park, Mofi Rahman, Mohit Khatwani, Natalie Dao, Nenshad Bardoliwalla, Nesh Devanathan, Neta Dumai, Nilay Chauhan, Oscar Wahltinez, Pankil Botarda, Parker Barnes, Paul Barham, Paul Michel, Pengchong Jin, Petko Georgiev, Phil Culliton, Pradeep Kuppala, Ramona Comanescu, Ramona Merhej, Sara Mc Carthy, Sarah Cogan, Sarah Perrin, Sébastien M. R. Arnold, Sebastian Krause, Shengyang Dai, Shruti Garg, Shruti Sheth, Sue Ronstrom, Susan Chan, Timothy Jordan, Ting Yu, Tom Eccles, Tom Hennigan, Tomas Kocisky, Tulsee Doshi, Vihan Jain, Vikas Yadav, Vilobh Meshram, Vishal Haself, Warren Barkley, Wei Wei, Wenming Ye, Woohyun Han, Woosuk Kwon, Xiang Xu, Zhe Shen, Zhitao Gong, Zichuan Wei, Victor Cotruta, Phoebe Kirk, Anand Rao, Minh Giang, Ludovic Peran, Tris Warkentin, Eli Collins, Joelle Barral, Zoubin Ghahramani, Raia Hadsell, D. Sculley, Jeanine Banks, Anca Dragan, Slav Petrov, Oriol Vinyals, Jeff Dean, Demis Hassabis, Koray Kavukcuoglu, Clement Farabet, Elena Buchatskaya, Sebastian Borgeaud, Noah Fiedel, Armand Joulin, Kathleen Kenealy, Robert Dadashi, and Alek Andreev. Gemma 2: Improving open language models at a practical size, 2024. URL https://arxiv.org/abs/2408.00118.

Hugo Touvron, Louis Martin, Kevin Stone, Paul Albert, Abdullah Almahairi, Yacine Babaei, Nikita Bashlykov, Suchin Batra, Priya Bhargava, Shagun Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *CoRR*, abs/2307.09288, 2023. doi: 10.48550/ARXIV.2307.09288. URL https://doi.org/10.48550/arXiv.2307.09288.

Laurens van der Maaten et al. The Llama 3 herd of models. 2024. URL https://arxiv.org/abs/2407.21783.

Shubham Vatsal and Harsh Dubey. A survey of prompt engineering methods in large language models for different NLP tasks, 2024. URL https://arxiv.org/abs/2407.12994.

Liang Wang, Nan Yang, Xiaolong Huang, Linjun Yang, Rangan Majumder, and Furu Wei. Multilingual E5 text embeddings: A technical report. *arXiv*, 2024.

Junjie Ye, Xuanting Chen, Nuo Xu, Can Zu, Zekai Shao, Shichun Liu, Yuhan Cui, Zeyang Zhou, Chao Gong, Yang Shen, Jie Zhou, Siming Chen, Tao Gui, Qi Zhang, and Xuanjing Huang. A comprehensive capability analysis of GPT-3 and GPT-3.5 series models. *CoRR*, abs/2303.10420, 2023. doi: 10.48550/ARXIV.2303.10420. URL https://doi.org/10.48550/arXiv.2303.10420.

Hao Yu, Zachary Yang, Kellin Pelrine, Jean Francois Godbout, and Reihaneh Rabbany. Open, closed, or small language models for text classification?, 2023. URL https://arxiv.org/abs/2308.10092.

Appendix

A Metrics

A.1 Classifier metrics

Accuracy refers to the ordinary precision of predictions, and coverage refers specifically to the number of classification non-refusals. A refusal is a special type of prediction that shows it is better to transfer the client's request to a hired specialist. Let's call this special class *"unk"* (unknown). The coverage formula is:

$$\text{Coverage} = \frac{1}{N}\sum_{i=1}^{N}\mathbf{1}\{y_{C_i} \neq \text{unk}\},$$

where y_{C_i} is classifier prediction. The metric can also be used in annotation.

A.2 Annotation metrics

In the annotation task, the only effective way to assess an LLMs annotation capability is to measure how often its predictions coincide with those of human annotators, since the models initially have access only to human-provided labels. Confidence in human annotation can be increased by collecting labels from several annotators on the same instance and retaining it only if they all agree, thus forming a high-quality test set. Let N be the total number of samples, y_i be the prediction of annotator H (human) or M (LLM). Since human annotation on the benchmark was collected only twice (data collection from experts and data collection from regular annotators), we will fix H.
The accuracy of the LLM annotator M is:

$$\text{Accuracy}_H(M) \;=\; \frac{1}{N}\sum_{i=1}^{N}\mathbf{1}(y_i^{(M)} = y_i^{(H)}),$$

where $\mathbf{1}(\cdot)$ is an indicator function that equals 1 if $y_{M_i} = y_{H_i}$, and 0 otherwise.

Finally, we can combine accuracy and coverage into a single overall metric:

$$\text{Accuracy}_{\text{total}}(M) = \text{Accuracy}(M) \times \text{Coverage}(M),$$

which measures the fraction of instances that are both non-abstained and correctly labeled. **We can also treat the *"conf"* class ("multiple intentions") as equivalent to *"unk"* for LLMs metric calculation, since it appears only in experimental human annotations**. In binary annotation tasks, the usual F1 score (macro avg., because both classes are equally important), precision, and recall remain appropriate.

B Alignment (SFT)

Two approaches were used to improve annotation quality: soft fine-tuning on the classification task (adding a classification layer directly to the LLM) and Soft Fine-Tuning (SFT) to enhance the LLM reasoning ability for correct annotation. To avoid modifying all LLM parameters due to computational resource constraints, we employ a low-rank adaptation (LoRA) Hu et al. (2021)

B.1 SFT reasoning

To enhance qualitative reasoning in base LLMs, supervised fine-tuning (SFT) can be applied using high-quality labeled reasoning traces from human experts or advanced closed-source models. This approach improves annotation quality for both text-based and probabilistic methodologies, as SFT optimizes task representation even without explicit reasoning generation. Section D.4 empirically confirms that SFT also elevates annotation metrics

for probabilistic approaches. **Data for SFT (reasoning):** 8,000 (Binary, Training data with approximately equal distribution of the number of intents) and 2000 (Binary, Validation data with prod distribution of intents).

C Experiment setup

C.1 LLMs

LLMs can be used with a text-based approach or a prob-approach, with or without RAG, and can be fine-tuned using LoRA. List of LLMs in Section I

C.2 RAG

A combination of retrieval and reranking, selection of the number of documents to add to the prompt, combination with a prob-approach were conducted. Main parameters reviewed: number of documents appearing in the prompt, document and query similarity thresholds, but only the number of documents is of particular value.
Experiments:
1. Selecting documents relevant to the client's query sent directly to the chat (Table N.3).
2. Selecting documents relevant to the verifiable intent or intents in the annotation (Table N.4). Since documents are searched not only at the client's request, but also relevant to each of the intent options (in fact, this is done in pre-set), LLM receives more context.

C.3 Alignment experiments (SFT)

For SFT, LoRA with Llama3.3-70b (for classification tasks) and Qwen2.5-32b (for reasoning soft fine-tuning) were used. The main parameters (rank, alpha) were reviewed for LoRA. Chosen attention projections: $\mathbf{q_{proj}}$, $\mathbf{k_{proj}}$, $\mathbf{v_{proj}}$, $\mathbf{o_{proj}}$

C.3.1 SFT (reasoning)

The collected training data using GPT-4o OpenAI (2024) and Deepseek-R1DeepSeek-AI et al. (2025) were used to SFT the Qwen2.5-32b model. The reasoning of other Large Language Models was parsed further, with the reasoning placed in the <**think**></**think**> block and the answer placed in <**answer**></**answer**>. Generations were only used for training if they led to the correct answer (the LLM answer matched the answer of professional annotators with extensive experience). It's also important not only to be able to reason well about the question, but also to be able to work with the information given in the task. **Therefore, documents are also submitted in the prompt using RAG** (Section D.3) in the <**retrieved**></**retrieved**> block. It is important to note that the training data does not overlap with the benchmark data.

C.4 Methodological notes

See Section L

D Results

D.1 Vanilla comparison of LLMs

The results of the benchmark measurements are shown in Table N.1. During the experiments, it was decided to focus further work on two models, as they showed high and stable metrics, good instruction following, and an interesting contrast between coverage and accuracy: Llama3.3-70b and Qwen2.5-32b. Llama3.3-70b shows high coverage with lower accuracy, indicating that it avoids *"unk"* more often and is less uncertain, unlike Qwen2.5-32b, which selects intents more often when confident and returns unk when uncertain.

From the beginning of the experiments, it's clear that human annotators are more accurate in multi-class annotation tasks than LLM.

D.2 Prob-approach after reasoning

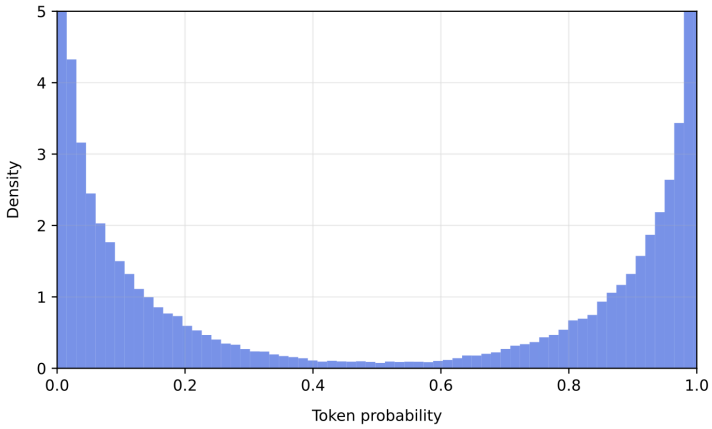

Figure 2: Distribution of response probabilities without reasoning

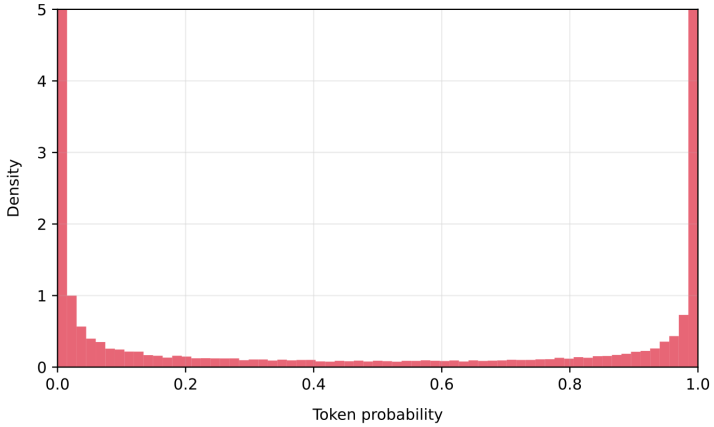

Figure 3: Distribution of response probabilities with reasoning

Assume a binary task (0 = "no", 1 = "yes"). We first run the text-approach, then delete the final answer in the `[ANSWER]` block, leaving only the rationale. Feeding this rationale back to the model with a single-token prompt yields a *prob-after-text* method that still supports thresholding.

The baseline distribution (Fig. 2) is U-shaped with peaks at 0 and 1 - clear separation and many confident cases. After injecting the rationale (Fig. 3), mid-range probabilities shrink as examples migrate to the extremes; near $1–\epsilon$ the model now forces a choice between opposite labels, so thresholding rarely triggers. In practice the model looks "more confident" even when its reasoning may be flawed. As Table D.2 shows, hiding the answer and thresholding 53% of low-confidence cases (*"unk"*) yields metrics similar to the text-approach and still below the pure probability method. Supplying a rationale therefore undercuts the main advantage of the probabilistic approach - flexible, effective rejection - without boosting accuracy.

MODEL	ACC.	F1-SCORE (MACRO. AVG.)	COVERAGE (PERC.)
QWEN2.5-32B TEXT-APPROACH	0.8059	0.7543	100%
QWEN2.5-32B PROB-APPROACH USE THRESHOLDS	**0.8530**	**0.8277**	54%
QWEN2.5-32B TEXT-APP.+PROB-APP	0.7971	0.7482	100%
QWEN2.5-32B TEXT-APP.+PROB-APP. USE THRESHOLDS	0.7646	0.7612	47%

Table 3: Metrics for prob-approach after text-approach (Binary)

D.3 RAG

D.3.1 Multi-class

Table N.3. With Llama-3-70B + *multilingual-e5* retrieval, fetching 5 docs raises coverage but trims accuracy, so $\text{Accuracy}_{\text{total}}$ stays almost unchanged (0.2735 → 0.2834). BM25 shows the opposite: 5 docs boost accuracy, coverage slips, and the total metric is again flat. Adding the *bge-m3* reranker to either retriever consistently lifts at least one metric and yields the best overall score for Llama-3-70B (0.2925 vs. 0.2712).

Table N.4. Retrieving docs for every intent enlarges the prompt (1–5 extra snippets per label), which invites hallucinations and inconsistent evidence, so $\text{Accuracy}_{\text{total}}$ drops.

D.3.2 Binary

In the case of binary annotation (Table N.5), adding the best RAG approach from multi-class annotating increases the annotation metrics for Llama3.3-70b prob-approach except for a slight drop in precision on positives (probably due to the strong increase in precision on negatives). Qwen2.5-32b also shows a significant increase in annotation metrics, but a severe drop in recall on negatives. We should also notice an increase in coverage for both LLMs, especially Qwen2.5-32b, which shows that adding more information increases the LLM's confidence in the answer while not decreasing the other metrics.

D.4 Result SFT (reasoning)

As shown in Table K.3, the metrics improved not only on the text-based approach (which is the original training goal), but also on the prob-approach. Interesting how much the recall on negatives has increased in the probabilistic approach. If before for good recall on negatives reasoning from LLM in text-approach mode, now on probabilistic approach a comparable recall on negatives is obtained. High-quality reasoning has indeed improved LLM metrics in the annotation task. It's also important to remember that RAG documents were added to the prompt during retraining. The improvement in metrics can also be attributed to the correct handling of retrieved documents, which Qwen2.5-32b learned from Deepseek-r1 and GPT-4o.

E Prob-approach

F Additional documents and benchmarks

Some client intents can be resolved using prepared documents, employed by support staff to answer client queries. Document length averages 500 characters, extending up to 2,000

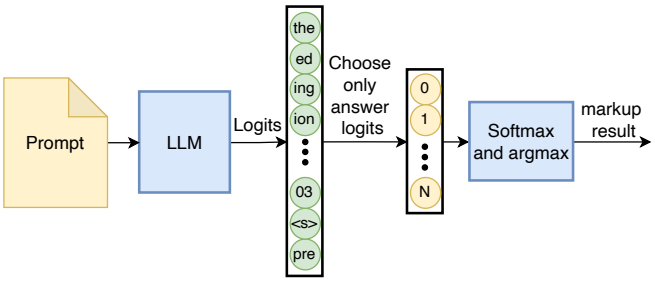

Figure 4: Prob-approach

characters if necessary. It's also important to note that the content of some documents may overlap or even conflict each other in some cases. A trained specialists manage these cases effectively. **Binary benchmark**: 18,000 (N-rows, Annotations of binary classifier predictions ("yes"/"no") for text classification correctness). **Multi-class benchmark**: 12,000 (N-rows, Selections from top-5 predicted classes of a text classifier).

To verify the quality of LLM annotation, quality annotations were collected from **highly qualified, expensive experts**. Then these data were marked up by the company's regular annotators, who currently annotate the predictions of the text classifier, as well as by LLM annotation. Due to the high cost and lengthy annotation process required by expert annotators, their services cannot be used for regular annotation. The number of unique classes is approximately 250 for benchmarks, with classes distributed according to product distribution.
The main criterion indicating that LLM annotation performs better than human annotation is: **all metrics obtained by LLM annotation must be equal to or better than regular human annotations when compared to expert benchmark annotations**. Despite this stringent requirement, the impact of possible annotation errors on classifier performance has also been studied (Section L.4), considering scenarios where solutions might reduce certain classification metrics.

G Description of prompts

G.1 Text-approach prompt (reasoning)

In order for the LLM to follow the instructions a prompt was written consisting of the following blocks: **Domain introduction.** Some information is given about the company, the clients, why it's important to do accurate annotation and check answers. **Data details.** The concept of intent is explained, as well as the fact that they come from a text classifier. **Task.** Depending on the modifications, the task includes at least the client's text and a description of the intent that needs to be checked to see if it fits the class, or a description of several intents from which only one suitable description needs to be selected. Next, relevant documents can be found (Section D.3) for the client's text, which may contain the answer to their question, explanations of terminology, and other useful information, as well as high-quality examples (positive and negative) of texts for intents. **Input and output format.** The LLM receives the task in the **[USER]** block. Then all reasoning should be done in the **[REASONING]** block and the final answer should be given in the **[ANSWER]** block. In case of binary annotation task, the final answer is 'yes' or 'no', in case of multi-class annotation the final answer is one selected intent from the list passed in the task. **Reasoning.** The prompt mentions that reasoning should go step by step, reveal the meaning of unfamiliar words in the client's request, use all the information provided, but not imagine it for the client.

G.2 Prob-approach prompt

The difference from the text-based prompt is that the prob-approach does not require an emphasis on reasoning. Also, after the instruction and task are given, before getting the probabilities of answers, the **[ANSWER]** block is added to the LLM response block (chat format depends on the specific LLM), therefore it's enough for LLM to generate only 1 character, which can be either 0 or 1 (if binary annotation), or a number from 1 to 5 (in the case of multi-class).

H Annotation completion time

Previous experiments conducted by the company showed that 1 multi-class annotation takes **13.4** seconds on average, while 1 binary annotation takes **6.8** seconds on average (these statistics take into account that several annotators can annotate tasks in parallel). When using vLLM Kwon et al. (2023) and 1 GPU per 1 multi-class, the best approach with the ensemble takes **0.85** seconds, which is **8 times faster** than the binary annotation task and **15.7 times faster** than multi-class annotation.
For example: multi-class benchmark took a human annotator almost two days to annotate, while LLM annotation took only 3 hours. The problem of annotation duration is particularly acute when several parallel annotations are running in a company. The ability to run LLM annotation on multiple GPUs **reduces the queue and speeds up the entire annotation process.**

I LLMs

- Llama3.3-70b van der Maaten et al. (2024)
  Dense decoder-only Transformer (RMSNorm+SwiGLU) with 128k-token vocabulary, pre-trained on 15T multilingual tokens and post-trained with SFT + Rejection Sampling + Direct Preference Optimisation.
  Native capabilities: multilingual Q&A, coding, reasoning and tool-use hooks.
- Qwen2.5-32b Qwen Team (2024)
  32b dense Transformer with revamped tokenizer and 18T-token pre-training corpus Model is purely dense, making it drop-in for GPU inference. Training focus: cold-start data + reinforcement learning that explicitly rewards chain-of-thought quality, bringing reasoning scores close to OpenAI-o1.
- Vikhr NeMo-12b Nikolich et al. (2024)
  Bilingual Russian-English instruction Mistral-Nemo NVIDIA Corporation & Mistral AI (2024) with a 40k SentencePiece vocab adapted to Russian.
  Pipeline: rebuild tokenizer, 11b-token continued pre-training with KL regularisation to avoid catastrophic forgetting, instruction tuning.
- Gemma2-27b Team et al. (2024)
  Decoder-only Transformer with Grouped-Query Attention and interleaved local-global attention, GeGLU activations and RMSNorm throughout.
- Athene-V2-Chat-72b Nexusflow (2024)
  Fine-tuned from Qwen2.5 72b with a "targeted post-training" RLHF pipeline; released under open weights by Nexusflow.
  Specialties: strong function-calling agent variant and long-log extraction accuracy, aimed at enterprise tool-use scenarios.

J RAG

J.1 Retrievals

- multilingual-e5-large-instruct Wang et al. (2024)
  Instruction-Tuned Multilingual Embedder supports 100 languages, built on XLM-RoBERTa-large. It's fine-tuned with an instruction format.

- BM25 Robertson & Zaragoza (2009)
  Okapi BM25 is a classic ranking function for document retrieval based on term matching. It scores documents by term frequency in the document and inverse document frequency of query terms, with length normalization. This yields a bag-of-words relevance score for a given query.

J.2 Rerankers

- bge-m3 Chen et al. (2024)
  Multi-Function Multilingual Embeddings is a Beijing Academy of AI model emphasizing Multi-Functionality, Multi-Linguality, and Multi-Granularity. The model can produce BM25-like term weights alongside embeddings, enabling hybrid search with no extra cost.
- FRIDA Snegirev et al. (2024)
  FRIDA is a general-purpose text embedding model inspired by a T5 denoising encoder.

J.3 Embedding storage

Document embeddings are stored in the faiss IndexFlatIP Douze et al. (2024). This index stores all vectors in a "flat" form, without additional structure or compression, and when searching, simply calculates the scalar product between the query and each vector in the index. This approach guarantees accurate search for the closest neighbors according to the scalar product metric.

K SFT metrics

K.1 Classification

To further improve annotation metrics, we fine-tune the LLM directly on the annotation task. First, annotation data are collected; then the LLM is modified to function as a classifier by adding a linear layer (classification head). The hidden state of the LLM's last token is fed into this head, after which the resulting LLM–classifier is trained like a standard classifier. All original LLM weights remain frozen except for the LoRA parameters and the classification head weights. This approach restricts the LLMs reasoning abilities but yields stable, high-quality annotation metrics, which is advantageous for automation.

DATASET	N-ROWS	DESCRIPTION
MULTI-CLASS TRAINING DATA	50,000	TRAINING DATA WITH APPROXIMATELY EQUAL DISTRIBUTION OF THE NUMBER OF INTENTS
MULTI-CLASS VALIDATION DATA	20,000	VALIDATION DATA WITH PROD DISTRIBUTION OF INTENTS

Table K.1: Data for SFT (classification)

As can be seen in Table K.1, training and validation data were collected from the production data stream, but in the training sample, the number of classes was balanced.

MODEL	COVERAGE (*"conf"="unk"*)	COVERAGE (*"conf"* $\neq$ *"unk"*)	ACCURACY (*w/o "conf"*)	ACCURACY (*"conf"=wrong*)	$\text{ACCURACY}_{\text{total}}$
HUMAN	0.2134	0.5240	**0.6977**	0.3113	0.1489
LLAMA3.3-70B TEXT-APPROACH	0.6585	0.6585	0.4118	0.4118	0.2712
LLAMA3.3-70B TEXT-APPROACH BEST RAG	0.6446	0.6446	0.4519	**0.4519**	**0.2913**
LLAMA3.3-70B LORA R=4, ALPHA=8 (1 EPOCH)	0.6712	0.6712	0.4055	0.4055	0.2722
LLAMA3.3-70B LORA R=4, ALPHA=8 (2 EPOCHS)	**0.7328**	**0.7328**	0.3455	0.3455	0.2532

Table K.2: Metrics after SFT (classification) (Multi-class)

As a result of training (Table K.2), the model began to choose unk more rarely, which led to an increase in coverage but a decrease in accuracy on all other intents. It was also found that the probability distribution had shifted in the direction described in Section D.2, which also made the thresholds more difficult to choose. RAG provides better total accuracy than this type of adaptation. Based on the results obtained on the multi-class data and the impossibility of obtaining explanations from the model using this approach, experiments on binary annotation were conducted using the method described in Section B.1.

K.2 Reasoning

MODEL	PREC. *1, pos.*	PREC. *0, neg.*	REC. *1, pos.*	REC. *0, neg.*	ACC.	F1-SCORE (MACRO AVG.)	COVERAGE (PERC.)
HUMAN	0.8748	0.7162	0.7834	0.8297	0.8018	0.7977	100%
QWEN2.5-32B TEXT-APPROACH BEST RAG	0.8787	0.6932	0.7591	0.8386	0.7904	0.7868	100%
QWEN2.5-32B TEXT-APPROACH LORA *r=4, alpha=8 dropout=0.1* BEST RAG	0.8947	0.7058	0.7565	0.8677	0.8012	0.7991	100%
QWEN2.5-32B PROB-APPROACH USE THRESHOLDS BEST RAG	0.8993	**0.8401**	**0.9712**	0.5818	**0.8908**	0.8103	59%
QWEN2.5-32B PROB-APPROACH USE THRESHOLDS LORA *r=4, alpha=8 dropout=0.1* BEST RAG **BEST LORA**	**0.9131**	0.7641	0.8131	**0.8867**	0.8430	**0.8405**	63%

Table K.3: Metrics after SFT (reasoning) (Binary, Prob-approach)

L Methodological notes

L.1 Parameters

All parameters were searched through on separate samples without test data leakage or adaptation to the test sample. To select the parameters, the load data was collected from production and annotated separately; the benchmark was used exclusively for final comparisons.

L.2 Completeness of experiments

If algorithms in one type of annotation (binary or multi-class) show a clear advantage in experiments, they're not repeated for another type of annotation, with some exceptions. The reason for this is to reduce the amount of computation.

L.3 Threshold search policy for a prob-approach

The threshold values for the prob-approach were also chosen therefore the approach would show results higher than human annotators on annotation metrics, but not the highest possible, since rejecting classification logically reduces coverage and increases precision (because only examples that the LLM is confident about are annotated). Since people also tend to make mistakes, if the annotation metrics are higher for LLM, the classification metrics will not only be closer to the true values when periodically measuring the quality of the classifier, but the trained classifier will also perform better on new data (Section L.4).
It may be possible to achieve even better results by combining approaches that were not described in detail in the experiments, or by carefully tuning all hyperparameters, but the

main goal of the experiments was achieved: the possibility of replacing humans with LLM without loss of quality in the annotation task was demonstrated.

L.3.1 RAG design

RAG (Retrieval Augmented Generation) typically comprises two components: retrieval and re-ranking. Retrieval selects a set of documents based on simple metrics (e.g., cosine similarity between encoder embeddings of the query and the documents). Although retrieval is fast, it does not order documents precisely by relevance. Retrieval can also be performed using statistical algorithms without neural networks (e.g., BM25 Robertson & Zaragoza (2009)). Retrieved documents are usually stored in a vector database. These documents can then be re-ranked by a more complex model (cross-encoders) which helps to select the most relevant documents and order them by importance to the query. Large documents are split into smaller ones using various techniques (by paragraph, special symbols, or sentence-similarity within a chunk). This chunking ensures each piece is small enough to include in full without further division. Relevant documents are placed in the prompt, after which the LLM solves the task in the usual format. The prompt instruction changes slightly, with the addition of the **[RETRIEVED]** block, explaining to the model that this block will contain information that may help to answer the client's query. In the task-block, after the description of the intents and examples, documents are added below the **[RETRIEVED]** block.

L.4 The impact of False Negative and False Positive on the classifier

L.4.1 False Positive

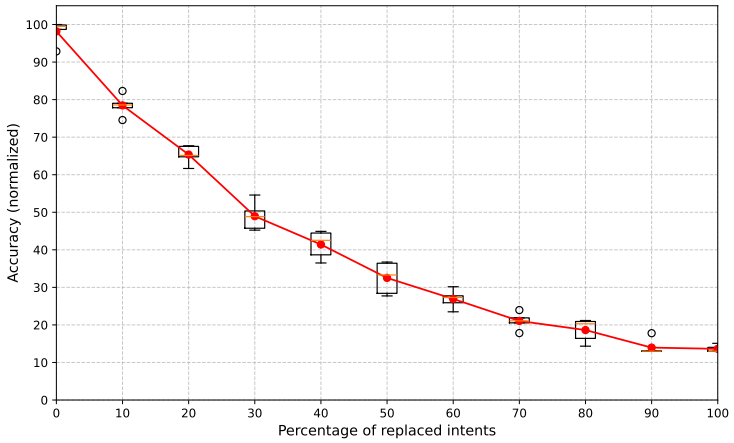

Figure 5: Impact of False Positive on the classifier (5 experiments)

If we replace the intent in the training sample with the closest similar one and train the text classifier on this, the quality of the test sample begins to deteriorate in a predictable manner. Starting at 70% replacements (Fig. 5), the quality begins to drop rapidly, but does not reach zero. This may indicate several things: **1.** One text in the data may correspond to several intents, therefore the closest replacements do not significantly impair the classifier's training. **2.** The marked data contains errors made by the annotators, and replacing it with the closest intent actually corrects these errors, there are enough corrected examples in the dataset for normal training.

L.4.2 False Negative

Removed examples (Fig. 6) in the annotation have a stronger impact on the accuracy of the classifier; accuracy drops faster when the dataset is affected than in the False Positive

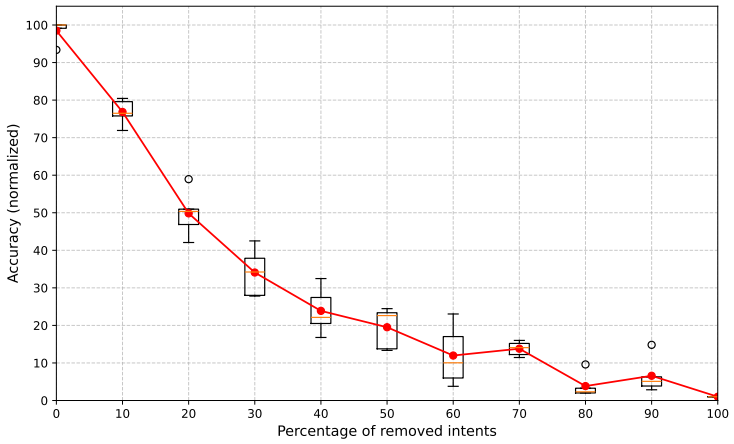

Figure 6: Impact of False Negative on the classifier (5 experiments)

example. It's likely that the lack of removed examples in intents with a small number of classes greatly affects the model's ability to learn them, which is why the problems become more noticeable in the test sample. It's obvious that with zero records in the training sample, no quality can be achieved. An experiment was also conducted to increase the proportion of *"unk"* in the training dataset, but this only resulted in the classifier starting with some percentage of replacements predicting exclusively unk.

M The importance of mistakes

After a series of experiments, it became clear that it was necessary to achieve a quality that was not inferior to human annotators across all selected metrics, since the classifier is affected by even the first observable errors in the dataset, and the impact on business metrics requires more in-depth analysis. If LLM can annotate data more accurately than human annotators (i.e., correctly select 1 out of 5 intents or check a specific intent for accuracy), then we can expect to improve the quality of the classifier itself when training on higher-quality data. Also, it is worth considering that reducing the number of False Negative has a more positive effect on the classifier.

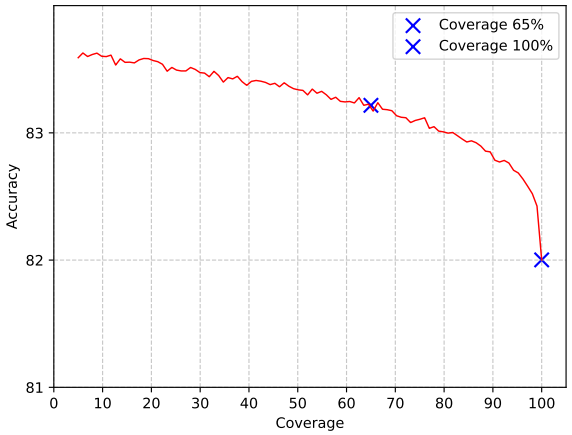

Figure 7: Example of accuracy dependence on coverage for a binary ensemble

N Full metrics

MODEL	COVERAGE (*"conf"="unk"*)	COVERAGE (*"conf" ≠ "unk"*)	ACCURACY (*w/o "conf"*)	ACCURACY (*"conf"=wrong*)	$\text{ACCURACY}_{\text{total}}$
HUMAN	0.2134	0.5240	**0.6977**	0.3113	0.1489
VIKHR NEMO-12B TEXT-APPROACH	0.6254	0.6254	0.4250	0.4250	0.2658
GEMMA2-27B TEXT-APPROACH	0.4688	0.4688	0.4931	0.4931	0.2312
ATHENE-V2-CHAT-72B TEXT-APPROACH	0.5498	0.5498	0.4001	0.4001	0.2200
LLAMA3.3-70B TEXT-APPROACH	**0.6585**	**0.6585**	0.4118	0.4118	0.2712
QWEN2.5-32B TEXT-APPROACH	0.5243	0.5243	0.5399	**0.5399**	**0.2831**

Table N.1: Metrics for LLMs (Multi-class)

MODEL	PREC. *1, pos.*	PREC. *0, neg.*	REC. *1, pos.*	REC. *0, neg.*	ACC.	F1-SCORE (MACRO AVG.)	COVERAGE (PERC.)
HUMAN	0.8748	0.7162	0.7834	0.8297	0.8018	0.7977	100%
LLAMA3.3-70B TEXT-APPROACH	0.8779	**0.8178**	**0.9743**	0.4600	**0.8711**	0.7562	100%
LLAMA3.3-70B PROB-APPROACH WITHOUT THRESHOLDS	0.8718	0.8067	0.9721	0.4496	0.8643	0.7483	100%
LLAMA3.3-70B PROB-APPROACH USE THRESHOLDS	0.8918	0.6825	0.7387	**0.8624**	0.7875	0.7850	57%
QWEN2.5-32B TEXT-APPROACH	0.8613	0.6531	0.8726	0.6307	0.8059	0.7543	100%
QWEN2.5-32B PROB-APPROACH WITHOUT THRESHOLDS	0.8471	0.6365	0.8837	0.5606	0.7977	0.7306	100%
QWEN2.5-32B PROB-APPROACH USE THRESHOLDS	**0.9398**	0.6863	0.8518	0.8560	0.8530	**0.8277**	54%

Table N.2: Metrics for LLMs (Binary)

MODEL	COVERAGE (*"conf"="unk"*)	COVERAGE (*"conf"* ≠ *"unk"*)	ACCURACY (*w/o "conf"*)	ACCURACY (*"conf"=wrong*)	$\text{ACCURACY}_{\text{total}}$
HUMAN	0.2134	0.5240	**0.6977**	0.3113	0.1489
QWEN2.5-32B	0.5243	0.5243	0.5399	**0.5399**	0.2831
LLAMA3.3-70B	0.6585	0.6585	0.4118	0.4118	0.2712
LLAMA3.3-70B *retriever: multilingual-e5-large-instruct* FIND 1 DOCUMENT	0.6479	0.6479	0.4221	0.4221	0.2735
LLAMA3.3-70B *retriever: multilingual-e5-large-instruct* FIND 5 DOCUMENTS	**0.7062**	**0.7062**	0.4013	0.4013	0.2834
LLAMA3.3-70B *retriever: bm25* FIND 1 DOCUMENT	0.6440	0.6440	0.4209	0.4209	0.2711
LLAMA3.3-70B *retriever: bm25* FIND 5 DOCUMENTS	0.6234	0.6234	0.4333	0.4333	0.2701
LLAMA3.3-70B *retriever: multilingual-e5-large-instruct reranker: FRIDA* FIND 5 FROM 25 DOCUMENTS	0.6416	0.6416	0.4517	0.4517	0.2898
LLAMA3.3-70B *retriever: multilingual-e5-large-instruct reranker: bge-m3* FIND 5 FROM 25 DOCUMENTS **BEST RAG**	0.6446	0.6446	0.4519	0.4519	0.2913
QWEN2.5-32B *retriever: multilingual-e5-large-instruct reranker: bge-m3* FIND 5 FROM 25 DOCUMENTS **BEST RAG**	0.5483	0.5483	0.5334	0.5334	**0.2925**

Table N.3: Metrics for RAG (Multi-class, text-approach)

Model	Coverage ("conf"="unk")	Coverage ("conf" ≠ "unk")	Accuracy (w/o "conf")	Accuracy ("conf"=wrong)	$\text{Accuracy}_{\text{total}}$
Human	0.2134	0.5240	**0.6977**	0.3113	0.1489
Llama3.3-70b	0.6585	0.6585	0.4118	**0.4118**	**0.2712**
Llama3.3-70b *retriever: multilingual-e5-large-instruct* Find 1 document	**0.6645**	**0.6645**	0.3669	0.3669	0.2438
Llama3.3-70b *retriever: multilingual-e5-large-instruct* Find 5 documents	0.6427	0.6427	0.3141	0.3141	0.2018

Table N.4: Metrics for RAG few-shot for intents (Multi-class, text-approach)

Model	Prec. *1, pos.*	Prec. *0, neg.*	Rec. *1, pos.*	Rec. *0, neg.*	Acc.	f1-score (macro avg.)	Coverage (perc.)
Human	0.8748	0.7162	0.7834	0.8297	0.8018	0.7977	100%
Llama3.3-70b text-approach	0.8779	0.8178	**0.9743**	0.4600	0.8711	0.7562	100%
Qwen2.5-32b text-approach	0.8613	0.6531	0.8726	0.6307	0.8059	0.7543	100%
Llama3.3-70b prob-approach use thresholds	0.8918	0.6825	0.7387	0.8624	0.7875	0.7850	57%
Qwen2.5-32b prob-approach use thresholds	**0.9398**	0.6863	0.8518	0.8560	0.8530	**0.8277**	54%
Llama3.3-70b *retriever: multilingual-e5-large-instruct reranker: bge-m3* Find 5 from 25 documents **best RAG** prob-approach use thresholds	0.8835	0.7416	0.7674	**0.8683**	0.8113	0.8106	61%
Qwen2.5-32b *retriever: multilingual-e5-large-instruct reranker: bge-m3* Find 5 from 25 documents **best RAG** text-approach	0.8787	0.6932	0.7591	0.8386	0.7904	0.7868	100%
Qwen2.5-32b *retriever: multilingual-e5-large-instruct reranker: bge-m3* Find 5 from 25 documents **best RAG** prob-approach use thresholds	0.8993	**0.8401**	0.9712	0.5818	**0.8908**	0.8103	59%

Table N.5: Metrics for RAG (Binary)

O Discussion

Two approaches combining LLMs with RAG were proposed: binary pipeline uses two LLMs (Llama3.3-70b, Qwen2.5-32b), and one of these (Qwen2.5-32b) was fine-tuned on the reasoning outputs of two large proprietary models. Considering annotation metrics (primarily accuracy and coverage), we replaced 65% of binary annotations with LLMs and all multi-class annotations with LLMs, accelerating annotation speed by an average factor of 8 while improving overall quality. Adapting an LLM purely as a classifier did not yield the same gains as fine-tuning an LLM for reasoning, although the reasoning-trained LLM also showed improved performance in the prob-approach - even when reasoning generation is disabled by design. The results of reasoning fine-tuning underscore the importance of correct initial prompt processing by the LLM and suggest that the model may ”know” the correct answer from the very start of its generation.
The focus of future work should be on: Threshold sensitivity. (Fig. 7); Abstention thresholds for *”unk”* were hand-tuned for each LLM and benchmark. Because token-probability calibration varies by model, prompt, and data distribution, these thresholds are unlikely to generalize across languages, models, or new datasets without ongoing monitoring and re-tuning.
Full distillation of the LLM. There may be a way to explore knowledge-distillation or structured pruning to compress the LLM while retaining cross-domain annotation accuracy.

These observations suggest that, while LLM-based annotation can match or exceed human performance in our setting, deploying similar pipelines elsewhere will require careful domain adaptation, re-ranking training, threshold calibration, and ongoing prompt and model optimization.

P Prompts

Multi-class prompt

Hello! Here are the instructions you should follow when communicating with me:
1. You are an intelligent data annotation assistant. You receive text that a user has written in the app's support chat, along with several hypothetical intents (classes) that this text may refer to. Along with the intent names, you will also receive a human-written description of the intent and typical examples, also selected by a human. Carefully read and compare the intent description and examples with the text that needs to be classified.
2. It is very important that you follow the instructions carefully and are very careful when making decisions.
3. For each correctly completed task, you will receive bonuses that you can spend on self-development and training.
4. If you are able to classify the text clearly, enter the correct class after the [ANSWER] tag. Use the exact class name that was provided in the list, as this is necessary for subsequent quality assessment.
5. The text may not belong to any of the provided classes or may not contain any intent at all. In this case, display the answer in the format [ANSWER] *"unk"* (*"unk"* number from the task list).
6. If the context of the user's message is insufficient to provide an unambiguous answer using the provided list of classes, display the answer in the format [ANSWER] *"unk"* (*"unk"* number from the task list). Please note that this intent should only be used in extreme cases when you find it difficult to choose between several intents. This is a rare situation; therefore, read the intent descriptions carefully and match them to the client's phrase.
7. Write and reason in Russian. If you find this difficult, do it as you see fit and then translate your thoughts into Russian.
8. When solving tasks, be sure to write down all your reasoning in the text and analyze it while writing your answer.
9. Stick to the required output format:
1. [TEXT] user's text
2. [REASONING] Reasoning regarding the selection of the appropriate class. Reason consistently, consider all possible options, and use the class description and examples provided for it.
3. [ANSWER] Answer - the number of one of the intents in the list. If you decide that the answer should be one of the intents in the list, make sure you answer **ONLY** with the number of that intent.
10. You may also encounter additional data enclosed in the [RETRIEVED] tag. This data was obtained from the internal information system and may contain additional useful information about the user's text.

Multi-class prob-approach prompt

Hello! Here are the instructions you should follow when communicating with me:
1. You are an intelligent data annotation assistant. You receive text that a user has written in the app's support chat, along with several hypothetical intents (classes) that this text may refer to. Along with the intent names, you will also receive a human-written description of the intent and typical examples, also selected by a human. Carefully read and compare the intent description and examples with the text that needs to be classified.
2. It is very important that you follow the instructions carefully and are very careful when making decisions.
3. For each correctly completed task, you will receive bonuses that you can spend on self-development and training.
4. If you are able to classify the text clearly, enter the correct class after the [ANSWER] tag. Use the exact number that was provided in the list, as this is necessary for subsequent quality assessment.
5. The text may not belong to any of the provided classes or may not contain any intent at all. In this case, enter your answer in the format [ANSWER] *"unk"* (*"unk"* number from the task list).
6. If the context of the user's message from [TEXT] is insufficient to provide a clear answer using the provided list of classes, enter your answer in the format [ANSWER]: *"unk"* (*"unk"* number from the task list). Please note that this intent should only be used in extreme cases when you find it difficult to choose between several intents. This is a rare situation; therefore, read the intent descriptions carefully and match them to the user's phrase.
7. Stick to the required output format: [ANSWER] Answer - number of one of the intents in the list. If you decide that the answer should be one of the intents in the list, make sure you respond with **ONLY** the number of that intent.
8. You may also encounter additional data enclosed in the [RETRIEVED] tag. This data was obtained from the internal information system and may contain additional useful information about the user's text.

Binary prompt

Hello! Here are the instructions you should follow when communicating with me:
1. You are an intelligent data annotation assistant. You receive text that a user has written in the application's support chat and the suggested class to which this text may belong. You need to determine whether the text actually belongs to the suggested class or not. In addition to the class descriptions written by a human, you will also receive typical examples, also selected by a human. Read carefully and compare the class descriptions and examples with the text that needs to be classified.
2. It is very important that you follow the instructions carefully and are very careful when making decisions.
3. For each correctly completed task, you will receive bonuses that you can spend on self-development and training.
4. Write and think in Russian. If you find this difficult, do it as you see fit and then translate your thoughts into Russian.
5. Please note that the task text contains both suitable and unsuitable examples. Carefully compare the text with the examples to avoid falling into a trap. It is also important to consider product names or additional external context. If a class refers to one product or situation, but the text mentions another product or situation, this means that the text does not fit the specified class.
6. When solving problems, be sure to write down all your reasoning in the text and analyze it while writing your answer. Reason step by step; for convenience, you can break your reasoning down into separate points.
7. Follow the required format for presenting your conclusions:
1. [TEXT] User's text
2. [REASONING] Reasoning used to verify the correctness of the prediction. Reason consistently, step by step, consider all possible options, and be sure to use the class description and examples provided for it. Also, clarify the meaning of ambiguous words that may imply that the text does not belong to this class. Do not guess what the client means or try to find hidden meanings in their message. Evaluate whether a message belongs to a class based *only* on the text of the message.
3. [ANSWER] The answer is a single line "yes" or "no" in lowercase (small letters). If the text belongs to the intended class or is similar to the examples provided, answer "yes." If the text cannot be related to this class and is not similar to any of the examples given, answer "no." Be sure to give your final answer based on your previous reasoning.
8. You may also encounter additional data enclosed in the [RETRIEVED] tag. This data was obtained from the internal information system and may contain additional useful information about the user's text.

Binary prob-approach prompt

Hello! Here are the instructions you should follow when communicating with me:
1. You are an intelligent data annotation assistant. You receive text that a user has written in the application's support chat and the suggested class to which this text may belong. You need to determine whether the text actually belongs to the suggested class or not. In addition to the class descriptions written by a human, you will also receive typical examples, also selected by a human. Read carefully and compare the class descriptions and examples with the text that needs to be classified.
2. It is very important that you follow the instructions carefully and are very careful when making decisions.
3. For each correctly completed task, you will receive bonuses that you can spend on self-development and training.
4. Please note that the task text contains both suitable and unsuitable examples. Carefully compare the text with the examples to avoid falling into a trap. It is also important to consider product names or additional external context. If a class refers to one product or situation, but the text mentions another product or situation, this means that the text does not fit the specified class.
5. Follow the required format for presenting your conclusions: 1. [TEXT] User's text, [ANSWER] The answer is a single line "0" or "1" in lowercase (small letters). If the text belongs to the intended class or is similar to the examples provided, answer "1." If the text cannot be related to this class and is not similar to any of the examples given, answer "0."
6. You may also encounter additional data enclosed in the [RETRIEVED] tag. This data was obtained from the internal information system and may contain additional useful information about the user's text.