BULLETIN of Cherkasy State Technological University

Journal homepage: https://bulletin-chstu.com.ua/en Vol. 30 No. 3. 2025

UDC 004.85:004.912
DOI: 10.62660/bcstu/3.2025.106

Continuous feedback loops: Online fine-tuning of LLMs with user signals

Sofiia Shvets*

Master of Science, Data Scientist
NinjaTech AI
94022, 4410 El Camino Real Str., Los Altos, United States of America
https://orcid.org/0009-0000-7896-6479

Abstract. The intensive growth in the use of real-time language models requires mechanisms for their dynamic adaptation to changes in queries, terminology, and user expectations. The study aimed to investigate approaches to continuous feedback-based retraining of large language models. To achieve this goal, the theoretical and structural-functional modelling of the adaptation architecture, experimental implementation of the language model retraining cycle with processing and classification of different types of feedback, and quantitative evaluation of the results using automatic and user metrics were applied. The results of the study showed the effectiveness of the architecture of continuous online learning, which ensures the relevance and stability of the language model in real time. The study determined that implicit feedback is 4-10 times more common than explicit feedback, but explicit feedback gives a higher increase in the accuracy of answers. The proposed system successfully integrated different types of user signals, providing dynamic generation of training examples and hybrid relearning while maintaining the quality and consistency of the results. The Python software cycle for adaptive retraining of the language model involved processing and filtering user signals to form a high-quality buffer of training pairs. After 500 retraining steps on 52,912 query-response pairs, a significant improvement of the model was observed, which was confirmed by a decrease in the loss function from 3.82 to 3.15 and stability of the fine-tuning process without signs of overtraining. The results of the pre-training showed a moderate improvement in the quality of answers after adaptation: lexical similarity according to the Recall-Oriented Understudy for Gisting Evaluation was 0.102, accuracy according to the Bilingual Evaluation Understudy was 0.006, and subjective user satisfaction increased to 0.24, while maintaining the stability of the model with an average cosine similarity value of 0.396. The approach proposed in this study improves the quality and relevance of real-time responses of language models while maintaining their stability and can be used in productive systems to improve user experience

Keywords: adaptive relearning; generative transformers; dynamic model adaptation; Python implementation; hybrid learning; quality assessment metrics; language model stability

INTRODUCTION

Large Language Models (LLMs) are a key component of modern Artificial Intelligence (AI) systems that operate in productive environments ranging from generative assistants and chatbots to search engines, recommender systems, and automated support platforms. However, after initial deployment, such models typically remain fixed, unable to adapt to new queries, changes in terminology, or user feedback. This static nature limits their relevance, reduces their accuracy, and limits the analysis of the dynamics of the environment in which the AI system operates. This is especially critical in real-time, when user signals are generated constantly but do not affect the behaviour of the model. This situation highlights the key problem with the current use of

Article's History: Received: 12.05.2025; Revised: 01.08.2025; Accepted: 15.09.2025.

Suggested Citation:
Shvets, S. (2025). Continuous feedback loops: Online fine-tuning of LLMs with user signals. *Bulletin of Cherkasy State Technological University*, 30(3), 106-120. doi: 10.62660/bcstu/3.2025.106.

*Corresponding author

LLMs: the lack of continuous learning mechanisms that integrate feedback into the adaptation cycle. It is this problem that necessitates the creation of systems capable of ensuring the evolution of the language model by changes in user needs, behaviour and expectations.

For example, D.M. Anisuzzaman *et al.* (2025) highlighted that fine-tuning can be used to adapt LLMs to specific tasks, significantly increasing the accuracy and relevance of answers. At the same time, they emphasised the limitations of the traditional approach, namely the fixedness of models after training, which prevents them from quickly adapting to new requests and changes in the environment. G. Balaskas *et al.* (2025) studied the impact of continuous pre-training on the ability of LLMs to follow instructions and proved that regular updating of the base model, even without manual fine-tuning, can keep it relevant. This directly highlights the need for dynamic model evolution to ensure alignment with changing user demands. Furthermore, S. Rehan *et al.* (2025) demonstrated that the integration of Human-in-the-Loop (HITL) into the LLMs' postgraduate education process ensures continuous adaptation cycles based on real-time expert feedback. The study emphasised that static models without such mechanisms lose their relevance in a dynamic codebase environment. The results of the study by S. Hao & L. Duan (2025) demonstrated that approaches such as Reinforcement Learning from Human Feedback (RLHF), which address the strategic behaviour of users in real time, provide more accurate aggregation of preferences. This avoids the mistakes associated with unreliable feedback, which is critical for dynamic relearning in productive systems.

T. Shi *et al.* (2024) proposed the WildFeedback framework, which automatically collects and classifies feedback from real user sessions, which is especially relevant in the context of solving the problem of "frozen" models. This approach provided continuous adaptation of LLMs to the needs of users without manual intervention, increasing the relevance of responses in real-world applications. N. Rawal *et al.* (2024) developed a framework for retraining LLMs based on heterogeneous types of feedback (both explicit and implicit), unifying them into a single format for Supervised Fine-Tuning (SFT) and RLHF. This solves the problem of inconsistency and low suitability of individual feedback fragments, which complicates traditional retraining. A.V. Cardó (2025) analysed alignment LLMs with human values through RLHF using multi-level feedback that transforms simple ratings into data for reward models.

J. Wang *et al.* (2024) proposed a hybrid hierarchical architecture that combines LLMs with classic recommendation models for dynamic discovery of new user interests. This approach demonstrates that the use of LLMs in productive systems requires constant updating and the generation of new interest clusters. X. Lin *et al.* (2024) developed an approach to online updating of LLMs with adaptive selection of examples of medium complexity, which improves the efficiency of use of feedback in RLHF mode. This highlights the importance of developing continuous retraining cycles that provide faster model adaptation through dynamic data selection. Furthermore, a study by M. Punnaivanam & P. Velvizhy (2024) showed that contextualised summarisation that considers user intent requires adaptive retraining of LLMs capable of adapting to changes in the environment and personal preferences in real time. The proposed framework uses lightweight adapters and query templates for dynamic personalisation, which addressed the problem of "frozen" models and confirms the need for continuous adaptation based on feedback.

Despite advances in certain aspects of LLMs adaptation, most existing works do not cover the full cycle of continuous retraining from feedback collection to model stability assessment. Instead, the current study implements a holistic system for adapting the Generative Pre-trained Transformer 2 (GPT-2) model based on real user feedback from the OpenAssistant dataset. The study aimed to analyse the mechanisms of integrating feedback into online LLMs, through architecture development, software implementation of the pipeline, and evaluation of the quality of adaptation.

MATERIALS AND METHODS

At the first stage of the study, the architecture of the adaptation cycle was implemented, which integrates user feedback into the learning process (Kampelopoulos *et al.*, 2025). The system of continuous online learning for LLMs included query processing, feedback collection, formation of a buffer of training examples, and fine-tuning of the model with stability control. A hybrid of SFT and RLHF with the ability to dynamically switch depending on the type of examples was used as an adaptation mechanism. The Dynamic Reliability Weighted Aggregation (DRWA) formula (1) is used to consider the reliability of feedback sources (Haider *et al.*, 2025):

$$\omega_t = w_t + \varepsilon, R_{DRWA} = \omega_t \times \mu_T + (1 - \omega_t) \times \mu_U, \quad (1)$$

where μ_T, μ_U – average values of signals from trusted and untrusted groups; ε – small increase in trust with low variance. The formula (2) for weighted feedback aggregation was used to integrate multiple assessments into a single generalised response (Dombi & Jónás, 2022):

$$\hat{y} = \frac{\sum_{i=1}^{n} w_i \times y_i}{\sum_{i=1}^{n} w_i}, \quad (2)$$

where $\hat{y}$ – aggregate score; w_i – weight of the i-th feedback signal; y_i – value of the i-th feedback signal; n – number of feedback signals. Adaptive retraining (3) was performed using the Step-wise Adaptive SFT and RL (SASR) algorithm, which dynamically scales the contribution of examples based on the gradient norm (Terras *et al.*, 2025):

$$\lambda_t = clip(\frac{\|g_t\|}{\|g_{warm}\|}, 0.1), \quad (3)$$

where $\|g_t\|$ – the current gradient norm (adaptively reflects the complexity and variability of a new training example); $\|g_{warm}\|$ – the gradient norm in the warm-up stage (the basic SFT stage); *clip* (*x*, 0.1) – a function to limit the value within (0, 1) to prevent excessive updates. The quality control of the model after retraining (4) was ensured using the Elastic Weight Consolidation (EWC) method (Next Electronics, n.d.):

$$L(\theta) = L_{new}(\theta) + \sum_i \frac{\lambda}{2} F_i\left(\theta_i - \theta^*_{A,i}\right)^2, \quad (4)$$

where F_i – parameter importance score θ_i (Fisher matrix); $\theta^*_{A,i}$ – parameter value after pre-training; λ – regularisation coefficient that controls the strength of the constraint. As part of the adaptation cycle, all user signals (explicit and implicit) were pre-processed, classified, aggregated, and buffered (Zhao *et al.*, 2018). The buffer was formed in the form of structured request-expected response pairs, with weights calculated according to the reliability of the signals. Ninja LLM Suite was considered as an example of a research software system that implements the principles of continuous adaptation of LLMs. This platform provides comprehensive processing of various types of user signals, their filtering and integration into the cycle of hybrid online training of language models, which can be used to maintain their relevance and stability in real time (NinjaTech AI, n.d.). For this stage, theoretical architecture modelling, structural and functional modelling, and mathematical formalisation methods were used.

At the second stage, a software cycle for retraining the GPT-2 model in Python using the OpenAssistant dataset (OASST1) was implemented (OpenAssistant conversations…, n.d.). English-language examples with positive moderation and a length of at least 5 words were selected from the corpus. For each pair, the weight was calculated using the rank field, and a buffer of 52,912 pairs was stored in the prompt response weight format. Further training was performed using the Hugging Face Transformers library. The study implemented 500 gradient update steps using Trainer, TrainingArguments, and DataCollator. At this stage, software modelling, experimental implementation, and parametric performance analysis were used.

At the third stage, a comprehensive testing and quantitative evaluation of the results of adaptive retraining of the GPT-2 model was performed. For this purpose, the automatic metrics Recall-Oriented Understudy for Gisting Evaluation Longest Common Subsequence (ROUGE-L) and Bilingual Evaluation Understudy (BLEU) were used to determine the similarity with the reference answers, as well as the User Experience (UX) metrics: User Satisfaction Score (USS) and Relevance. To visualise the effects of retraining, a graph of the loss function reduction, a chart with quality metrics based on 500 examples, a graph of USS improvement over time, a table of changes in model responses before and after adaptation with feedback weights, and a histogram of the cosine similarity distribution were created. The methodology of the stage included quantitative evaluation, content analysis, UX evaluation, and stability analysis of vector representations.

RESULTS

Architecture of continuous online learning LLMs and the model adaptation cycle

Modern LLMs have become widespread in various fields due to their ability to generate high-quality text based on a variety of queries. However, to efficiently process real-world productive systems, they need to accommodate dynamic changes in user preferences, usage context, and types of information being queried. These challenges require the development of flexible architectures that provide continuous adaptation of language models, increasing their relevance and accuracy in real time. One of the key features of LLMs used in productive systems is the need to constantly adapt to changing interaction conditions. Changes in query styles, the emergence of new topics, or the transformation of user preferences create a demand for models that can be dynamically updated without complete retraining. Static LLMs trained offline do not take these factors into account and lose relevance over time.

The solution is a continuous online learning architecture that integrates feedback signals into the main adaptation cycle. User signals, such as explicit (ratings, corrections) and implicit (viewing time, scrolling), are processed at multiple levels: classification, aggregation, buffering, and transformation into training examples. After accumulating a sufficient volume of relevant examples, the model is retrained using SFT or RLHF. All changes are monitored using quality, stability, and A/B testing metrics. After each retraining cycle, the updated model is saved as a separate version with a unique identifier, which can be used to track the changes made and, if necessary, revert to previous stable configurations. The architecture of the LLMs adaptation system, which illustrates the main cycle of "request response feedback retraining", is shown in Figure 1.

This diagram illustrates the architecture of continuous online learning for LLMs, where user queries are processed by the model, and the received feedback is classified, buffered, and transformed into training examples, which are used for re-training and quality control, followed by model updates. The architecture of continuous online learning for LLMs is based on the integration of user feedback into the model adaptation cycle, which can be used for keeping the generation of answers up-to-date in a dynamic environment. In the process of collecting user signals, the system records both explicit feedback (e.g., ratings, answer corrections) and implicit feedback (interaction time, scrolling). Each signal is classified (whether it is positive/negative or changes in tone) and aggregated into a representation

of the corresponding quality. For example, according to the DRWA methodology, the weight for trusted signals varies depending on their consistency (1) (Haider *et al.*, 2025). After classification and reliability assessment, all signals are accumulated in a buffer, where the data is stored until training examples are generated. This way, a set of user-rated information is accumulated, ready to be converted into a query-expected-response format. This approach ensures that signals with real meanings are included in the training, not just random noise.

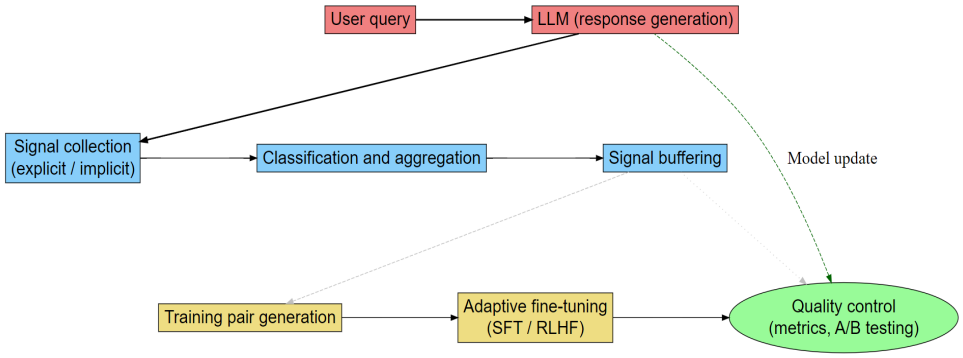

Figure 1. Diagram of the LLMs adaptive learning cycle
Source: compiled by the author based on D. Kampelopoulos *et al.* (2025)

At the stage of forming the data buffer for training, user signals are transformed into structured query-expected response pairs, which are the basis for further training of the language model. This process includes feedback aggregation, which addresses various user ratings and corrections. One approach to aggregation is to use weights that reflect the reliability and relevance of each signal (2) (Dombi & Jónás, 2022). In this case, the weights can be determined based on various criteria, such as source reliability, response time, or user interaction history. This reduces the influence of false or insignificant feedback and improves the quality of training data. After the data collection and buffering stages, the next step in the online learning architecture is the adaptive learning algorithm. A hybrid strategy is used that combines SFT and RL with human feedback (e.g., RLHF), with the ability to switch between them dynamically. In the corresponding SASR approach, the ratio between the proportions of SFT and RL at each step is determined by comparing the current gradient norm and the gradient norm after the initial "warm-up" phase (3) (Terras *et al.*, 2025).

Regarding quality control, after online retraining, it is crucial to prevent degradation of the overall model capability, especially to avoid catastrophic forgetting. One way to do this is to add a regulatory penalty for changing core parameters from the previous version of the model, most often using the Elastic Weight Consolidation (EWC) method (4) (Next Electronics, n.d.). In general, this approach ensures a balanced adaptation of new data without losing the knowledge already acquired, which is critical for the stable performance of the system. The successful functioning of adaptive relearning largely depends on the type and quality of the feedback received. A general breakdown of the types of user feedback used to adapt models is shown in Figure 2.

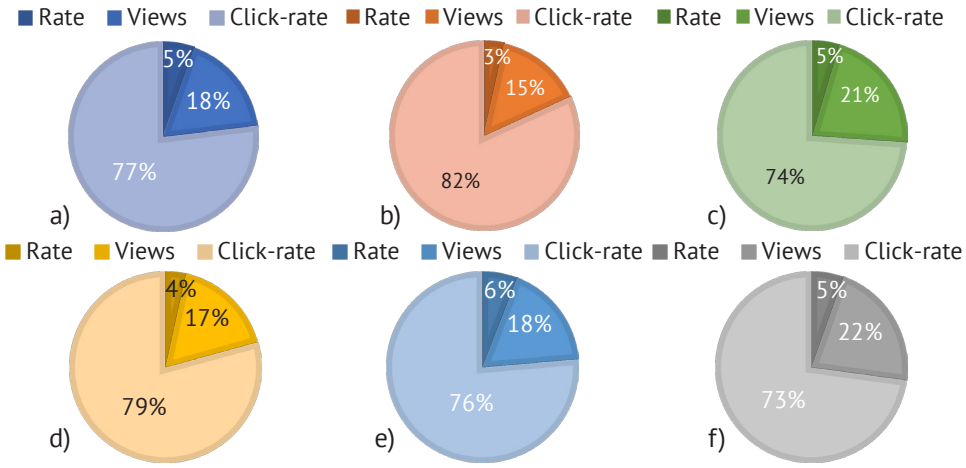

Figure 2. Distinguishing between explicit and implicit user feedback
Notes: a) Matrix Factorisation – Rating; b) Matrix Factorisation – Action; c) Bandit-Two; d) Bandit-Four; e) Bandit-State; f) Reinforce-State
Source: compiled by the author based on Q. Zhao *et al.* (2018)

These diagrams in Figure 2 show that implicit feedback (views, clicks, behavioural activity) in real systems is much more frequent than explicit feedback (ratings, preferences), which confirms its importance in the context of adaptive relearning. In particular, the average amount of implicit interaction (e.g., the number of views) for most models is 4-10 times higher than the number of explicit ratings. This indicates the need for improved classification and weight aggregation of implicit signals to effectively transform them into full-fledged training examples. In other words, explicit feedback provides a higher increase in relevance and consistency metrics compared to implicit feedback.

In general, the described architecture can be implemented both in a cloud-based environment and locally using containerised components. Its structure can be used for parallel data processing, independent model retraining, and automatic control over the update of the production version. This is in line with the engineering practices used in LLM adaptation systems, including NinjaTech AI. For instance, Ninja LLM Suite is a complex AI system that combines several advanced models and infrastructure optimisations to ensure maximum speed and accuracy of answers (NinjaTech AI, n.d.). This platform is designed to meet the requirements of continuous online learning, where different components work in synergy to instantly respond to changes in user requests and behaviour. Using customised, fine-tuned models such as Turbo 1.0 for quick responses and Apex 1.0 for deep analytical insights, Ninja LLM supports a wide range of applications from code generation to in-depth research. Thus, the continuous online learning architecture not only ensures that LLMs are updated in real time but also maintains their stability, relevance, and adaptability through integrated feedback, hybrid training, and systematic quality control.

Software implementation of adaptive LLMs

In modern LLM applications, adaptive relearning is becoming a key element in improving the quality of user experience. Considering user feedback in the process of continuous model training increases the relevance and accuracy of answers, as well as adapting the model to individual preferences. Technological solutions for such continuous learning require a combination of user signal processing methods, algorithms for generating training examples, effective fine-tuning procedures, and comprehensive quality control. The first step in this adaptation cycle is to create a high-quality training example buffer, i.e., to transform raw user signals (e.g., positive ratings, browsing time, corrections to answers) into structured query-expected answer pairs with appropriate weighting factors. Based on the open OpenAssistant dataset, a Python program procedure illustrating this preprocessing process was implemented in the file prepare_prompt_response_pairs.py (OpenAssistant conversations …, n.d.) (Fig. 3).

prepare_prompt_response_pairs.py > ...

```python
dataset = load_dataset("OpenAssistant/oasst1", split="train")

df = pd.DataFrame(dataset)

df = df[(df['lang'] == 'en') & (df['deleted'] == False) & (df['review_result'] == True)]

prompts = df[df['role'] == 'prompter']
id2prompt = dict(zip(prompts['message_id'], prompts['text']))

answers = df[df['role'] == 'assistant']
answers = answers[answers['parent_id'].isin(id2prompt.keys())]

train_data = []

for _, row in tqdm(answers.iterrows(), total=len(answers)):
    prompt_text = id2prompt.get(row['parent_id'], None)
    response_text = row['text']

    if prompt_text is None:
        continue
    if len(prompt_text.split()) < 5:
        continue
    if len(response_text.split()) < 5:
        continue
```

Figure 3. Code snippet for generating training pairs from the OpenAssistant dataset

Source: compiled by the author

To build training pairs, the OpenAssistant open multilingual dataset was used, from which only English-language messages with positive moderation were initially selected. Query-response pairs are formed from this dataset by matching messages with the roles of promoter and assistant, after which basic quality filters are applied (for example, a minimum text length of 5 words). Additionally, a weighting factor is calculated for each example that considers the relevance of the feedback, which correlates with the principle of aggregating user signals. This code covers the stages of collecting, classifying, and aggregating user signals, as well as primary data filtering, which is critical for building a high-quality training example buffer. As a result, the programme

generated 21,265 query-expected response pairs that met the quality criteria (English language, no deletions, positive feedback, minimum length) (Fig. 4). The processing of a large amount of data is efficient, which indicates that the methodology is ready for large-scale use in productive adaptive learning systems.

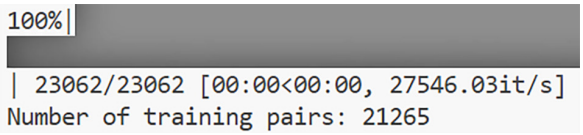

Figure 4. The result of the number of examples generated

Source: compiled by the author

As a result of executing the script for generating training pairs, examples in the format are saved to a separate file (train_examples.txt): Prompt the text of the user's query, Response the response of the language model, weight a weighting factor that reflects the quality or relevance of the response (for example, calculated based on rank) (Fig. 5). These examples demonstrate different types of queries and responses from a real corpus, which improves adaptability to different communication scenarios and the quality of generation. Saving such examples is a substantial step for data quality auditing, manual verification, and analysis of the distribution of weights corresponding to different levels of user feedback.

train_examples.txt

```
1  Prompt: Can you write a short introduction about the relevance of the term "monopsony" in economics? Please
2  Response: "Monopsony" refers to a market structure where there is only one buyer for a particular good or s
3
4  Recent research has identified potential monopsonies in industries such as retail and fast food, where a fe
5
6  Overall, the concept of monopsony is essential to understanding the dynamics of labor markets and the impac
7
8  References:
9  Bivens, J., & Mishel, L. (2013). The Pay of Corporate Executives and Financial Professionals as Evidence of
10 Weight: 1.0
11
12 ---
13
14 Prompt: Can you write a short introduction about the relevance of the term "monopsony" in economics? Please
15 Response: Monopsony is a market structure in which there is a single buyer in a market. In the context of l
16
17 Research has shown that the existence of monopsony power in labor markets can have negative effects on work
18 Weight: 0.5
```

Figure 5. Excerpt from the source file of training examples with queries, answers and weighting factors for fine-tuning the model

Source: compiled by the author

The second step involves retraining the GPT-2 language model based on the generated prompt and response buffer. The pre-processed pairs are tokenised, considering the delimiter between prompt and response, and then a set of labels is created for the text generation task. Quality filters were applied to the training set, including limiting the maximum length to 512 tokens and using only relevant examples. Fine-tuning was performed using the Hugging Face Transformers library with training parameters adapted to a limited number of steps (500), which corresponds to the scenario of rapid adaptation of the model to new feedback. The customised components, including Trainer, DataCollatorForLanguageModeling and adapted hyperparameters, provides efficient model updating while maintaining generation stability. The above sequence of actions is implemented in Python code (file fine_tune_gpt2_oasst1.py), a fragment of which is shown in Figure 6.

In the process of retraining the GPT-2 model based on 52,912 query-response pairs from the OASST1 dataset, 500 gradient update steps were performed (about 3.78% of one epoch). The number of retraining steps was limited to 500 for reasons of time and computing resources, which verified the performance of the pipeline and assess the loss dynamics at early stages without the risk of overtraining or resource overruns. When comparing the model before and after fine-tuning, a significant improvement in its performance was recorded after retraining in terms of the loss function, the initial value of which was about 3.82, indicating significant uncertainty in the model's predictions at the baseline. In the process of retraining for 500 steps, the loss steadily decreased to about 3.15, demonstrating better consistency between the generated answers and the expected ones. At the same time, the loss value fluctuated between

(2.60-3.63), indicating the stability of the fine-tuning process without catastrophic drops. This dynamic confirms the model's ability to adapt to new data based on user feedback, which increases the accuracy and relevance of answers compared to the initial version of GPT-2 without retraining. The results demonstrated the potential of online training with partial model updates, even with an incomplete epoch (Fig. 7).

```python
pairs = []
for ex in dataset:
    if ex["role"] == "assistant":
        parent_id = ex["parent_id"]
        if parent_id and parent_id in id_to_example:
            parent = id_to_example[parent_id]
            if parent["role"] == "prompter":
                prompt = parent["text"].strip()
                response = ex["text"].strip()
                if prompt and response:
                    pairs.append({"prompt": prompt, "response": response})

print(f"Total pairs for training: {len(pairs)}")
print("An example of a couple:", pairs[0] if pairs else "no matches found")

MODEL_NAME = "gpt2"
tokenizer = GPT2Tokenizer.from_pretrained(MODEL_NAME)
model = GPT2LMHeadModel.from_pretrained(MODEL_NAME)

if tokenizer.pad_token is None:
    tokenizer.pad_token = tokenizer.eos_token

train_dataset = Dataset.from_list(pairs)

def tokenize_function(examples):
    sep = tokenizer.eos_token if tokenizer.eos_token else ""
```

Figure 6. Code snippet for training the GPT-2 model based on the request and response buffer from the OpenAssistant dataset

Source: compiled by the author

```
Total pairs for training: 52912
An example of a couple: {'prompt': 'Can you write a short introduction about the relevance of the term "mono
psony" in economics? Please use examples related to potential monopsonies in the labour market and cite rele
vant research.', 'response': '"Monopsony" refers to a market structure where there is only one buyer for a p
articular good or service. In economics, this term is particularly relevant in the labor market, where a mon
opsony employer has significant power over the wages and working conditions of their employees. The presence
 of a monopsony can result in lower wages and reduced employment opportunities for workers, as the employer
has little incentive to increase wages or provide better working conditions.\n\nRecent research has identifi
ed potential monopsonies in industries such as retail and fast food, where a few large companies control a s
ignificant portion of the market (Bivens & Mishel, 2013). In these industries, workers often face low wages,
 limited benefits, and reduced bargaining power, leading to a situation where they are dependent on the empl
oyer for their livelihood. This dependence can result in further suppression of wages and a decline in worki
ng conditions.\n\nOverall, the concept of monopsony is essential to understanding the dynamics of labor mark
ets and the impact of market power on workers. Further research is needed to understand the extent and impac
t of monopsonies on the economy and to develop policies to address this issue.\n\nReferences:\nBivens, J., &
 Mishel, L. (2013). The Pay of Corporate Executives and Financial Professionals as Evidence of Rents in Top
1 Percent Incomes. Journal of Economic Perspectives, 27(3), 57-78.'}
Map: 100%|████████████████████████████████████████████████████████████████████████████| 52
912/52912 [04:54<00:00, 179.65 examples/s]
Map: 100%|████████████████████████████████████████████████████████████████████████████| 529
12/52912 [00:16<00:00, 3171.49 examples/s]
{'loss': 3.8244, 'grad_norm': 12.331202507019043, 'learning_rate': 4.5e-06, 'epoch': 0.0007559721802237678}
{'loss': 3.6094, 'grad_norm': 8.166468620300293, 'learning_rate': 9.5e-06, 'epoch': 0.0015119443604475356}
{'loss': 3.4598, 'grad_norm': 7.178922176361084, 'learning_rate': 1.45e-05, 'epoch': 0.0022679165406713033}
{'loss': 3.2002, 'grad_norm': 7.32305908203125, 'learning_rate': 1.9500000000000003e-05, 'epoch': 0.00302388
87208950713}
{'loss': 3.2883, 'grad_norm': 6.526586532592773, 'learning_rate': 2.45e-05, 'epoch': 0.0037798609011188387}
{'loss': 3.3314, 'grad_norm': 8.73172664642334, 'learning_rate': 2.95e-05, 'epoch': 0.004535833081342607}
{'loss': 3.3324, 'grad_norm': 7.506481647491455, 'learning_rate': 3.45e-05, 'epoch': 0.005291805261566375}
```

Figure 7. Initial GPT-2 retraining on OASST1: analysis of loss dynamics over 500 steps

Source: compiled by the author

Thus, the implemented online learning approach demonstrates effectiveness even with a limited number of update steps, providing a qualitative improvement of the model's performance in productive conditions. Compared to other models, it should be emphasised that a static model that does not consider new interac-

tion patterns quickly loses its relevance and is unable to adapt to changes in user behaviour or new types of queries. Offline training, while providing periodic model updates, has a significant delay between data collection and parameter updates, which reduces sensitivity to relevant feedback. In contrast, online training provides more flexible and rapid adaptation, as the model is updated directly based on recent user signals. This helps maintain the relevance of answers and improve the quality of generation in the face of dynamic user interaction. Thus, in contrast to static or periodically updated models, the proposed approach provides continuous improvement of the generative quality of LLMs following current user expectations, which is critical in real time.

Testing and evaluation of the results

In the process of adaptive relearning for LLMs, it is critical not only to ensure the technical implementation of the learning cycle but also to conduct a comprehensive evaluation of the results obtained. Such an evaluation determined how effectively the model has integrated new experiences based on user feedback, whether there has been an improvement in the content quality of answers, and whether the overall stability of the model's behaviour has been maintained. Comprehensive testing included both automatic metrics, such as ROUGE, BLEU, and F1, which reflect compliance with expected responses, and UX metrics such as USS and Relevance, which characterise usability and relevance from the user's point of view. One of the key indicators of the effectiveness of the retraining process is the behaviour of the loss function during learning epochs. Its gradual decline indicates that the model is getting closer to a generalised representation of training data, and the prediction error is decreasing. Figure 8 shows the dynamics of losses during the fine-tuning of the GPT-2 model based on the training buffer generated from user feedback.

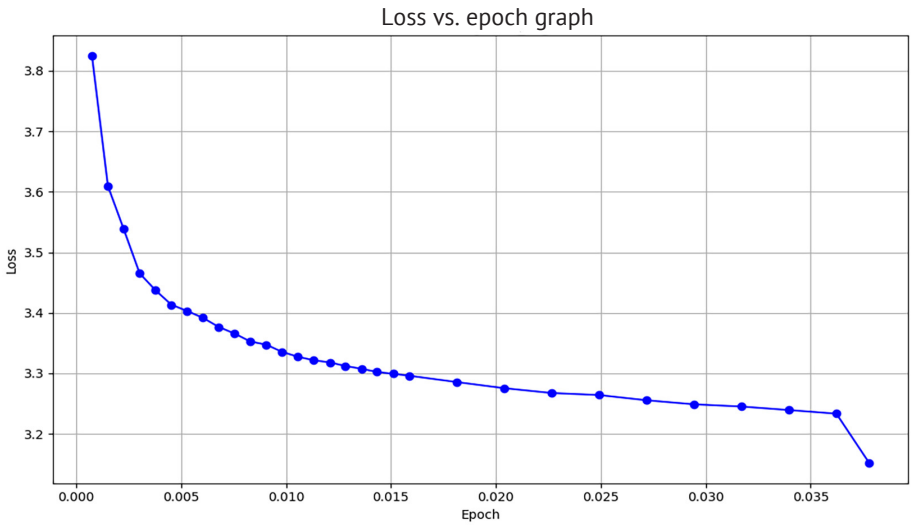

Figure 8. The dynamics of the loss function in the process of LLMs' retraining

Source: compiled by the author

The Figure 8 shows the change in the value of the loss function of the GPT-2 model during training on the generated buffer with user feedback. Each point corresponds to the loss value after a certain fraction of the training epoch. A fragment of the initial data is presented from 0.00075 (the beginning of the epoch) with a loss value of 3.8244 to 0.03779 with a loss value of 3.1522. The loss function steadily decreases with the number of steps, which indicates a gradual improvement in the model's ability to generalise patterns in training examples. This result confirms the correctness of the implemented pipeline retraining, and the effectiveness of the training data used.

In turn, the metrics used to evaluate the effectiveness of LLMs' pre-assignment are critical in quantifying the quality of the generated answers. Automatic metrics, such as ROUGE-L and BLEU, can be used to assess the lexical and semantic similarity between the generated response and the reference response, which reflects the accuracy of the content at the level of phrases and grammatical structures. ROUGE-L considers the largest common sequence of words, which is necessary for dialogues, while BLEU uses an n-gram approach. At the same time, automatic metrics do not always fully reflect the user's perception of the answer. That is why it is advisable to use UX metrics, in particular USS, which assesses the user's subjective satisfaction, and Relevance, which records the fact of a positive response to the model's answer. The combination of these metrics provides a comprehensive view of the effectiveness of adaptation, both in terms of linguistic consistency and practical usefulness for the end user. Figure 9 summarises the results of testing the GPT-2 model after retraining on 500 examples from the OpenAssistant dataset, which included real user queries and feedback responses.

```
🔢 Total pairs: 52912
100%|
 500/500 [55:39<00:00,  6.68s/it]
Downloading builder script: 5.94kB [00:00, ?B/s]
Downloading extra modules: 4.07kB [00:00, ?B/s]
Downloading extra modules: 3.34kB [00:00, ?B/s]

📊 Evaluation Results:
ROUGE-L: 0.1019
BLEU:    0.0060
USS (mean):       0.1883
Relevance (mean): 0.0000
📁 Results saved to eval_metrics_gpt2_oasst1.csv
```

Figure 9. The results of assessing the quality of GPT-2 model responses after adaptive retraining
Source: compiled by the author

This result incorporates the first 500 query-expected response pairs selected from 52,912 examples in the dataset, which was used to evaluate realistic processing time constraints. Answers were generated using the GPT-2 model after 500 gradient update steps. The calculated metrics were ROUGE-L = 0.1019, BLEU = 0.0060, USS (mean) = 0.1883, Relevance (mean) = 0.0000. The obtained values indicate a moderate improvement in content metrics with a limited number of pre-training steps, which suggests the potential for scaling the pre-training process by increasing the number of examples or adaptation cycles. To better demonstrate the effectiveness of adaptation, it is worth analysing the graph of USS improvement over time, which reflects the dependence of the quality of generations on the amount of feedback received (Fig. 10).

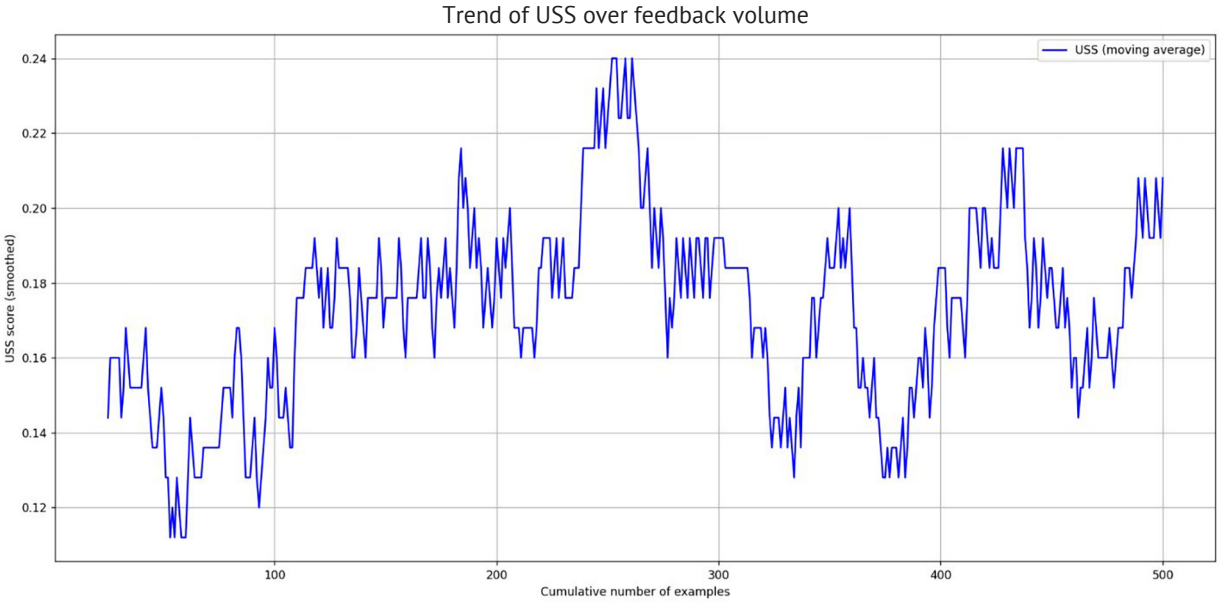

Figure 10. Dynamics of USS indicator improvement in the process of feedback accumulation
Source: compiled by the author

The results of retraining the GPT-2 model on OpenAssistant feedback show moderate but positive adaptation dynamics: despite the low average values of ROUGE-L (0.1019), BLEU (0.0060) and USS (0.1732), the USS plot shows an increase in the quality of generations with the accumulation of examples, reaching a local maximum of about 0.24. This indicates the potential for improvement with more data, better example filtering, and deeper adaptation, despite the initially zero Relevance score in the sample. An analysis of the examples of transformations of model responses before and after retraining is presented in Table 1.

Table 1. Analysis of changes in three model responses before and after adaptation based on user feedback

Request	Answer before	Answer after	Weight of feedback	Highlighted changes
What can be done...	Repeated phrase	Specific policies + authorities (FTC, DOJ, NLRB)	0.4	Qualitative clarification, without repetition

Continued Table 1.

Request	Answer before	Answer after	Weight of feedback	Highlighted changes
¿Cuál es el medio...?	Repeat the phrase	Answer with an example (aviation) and IATA.	0.6	Brevity, factuality
Y de todas ellas...	A meaningless phrase	Tailored advice for the user's interests	0.6	Content, personalisation

Notes: FTC – Federal Trade Commission; DOJ – Department of Justice; NLRB – National Labour Relations Board; IATA – International Air Transport Association
Source: compiled by the author

In other words, the model learned to better meet the expectations of users: to avoid repetition, to specify answers and to add meaningful details. In the first case, it replaced generalisations with clear policy measures with the names of the responsible authorities; in the second, instead of empty repetition, it provided a substantiated answer with a link to the source; in the third, it transformed a meaningless phrase into an advisory and personalised response. This indicates that the user's preferences are being assimilated, even with moderate feedback (0.4-0.6). Figure 11 shows the distribution of cosine similarity between the model's responses before and after adaptation.

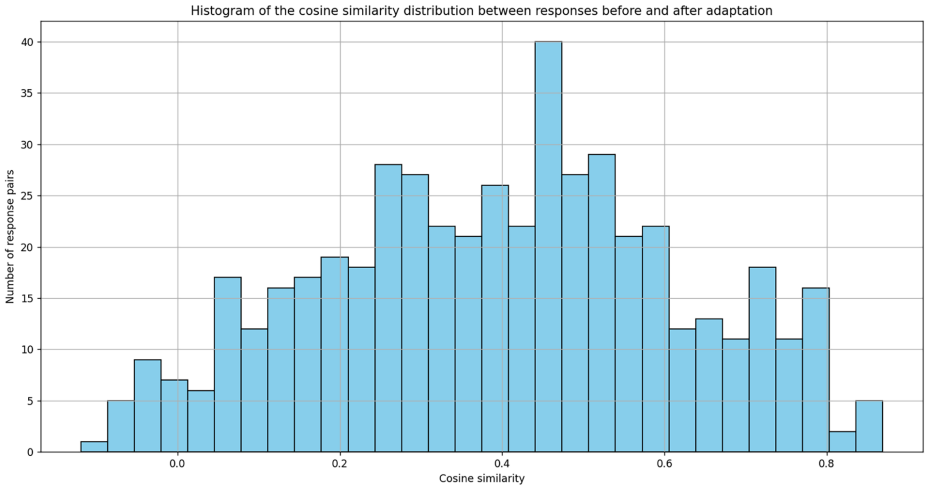

Figure 11. Histogram of cosine similarity distribution between model responses before and after adaptation
Source: compiled by the author

The results obtained indicate a moderate level of model stability after retraining: the average value of cosine similarity is 0.3964, which indicates a partial similarity between the vector representations of responses before and after adaptation. The value of the minimum similarity (-0.1191) indicates the presence of individual cases of significant response transformation, including potentially antagonistic changes. The maximum similarity of 0.8689 demonstrates that some responses have hardly changed significantly. This result indicates that the model, in the process of adaptation, simultaneously preserves the core of the formed knowledge while making sufficient changes to improve the quality of answers.

Thus, the evaluation results indicate a moderate but steady improvement in the quality of the model's answers after adaptive retraining based on user feedback. The model demonstrates the ability to preserve the basic cognitive structure of answers, while making meaningful refinements and unique changes that increase relevance and usefulness. The use of a comprehensive set of metrics and vector similarity analysis confirms the balance between adaptation and stability, which is key to effective online LLMs.

DISCUSSION

The study showed that continuous online training of LLMs with a hybrid SFT and RLHF approach based on the OpenAssistant dataset provides a moderate improvement in the quality of answers (ROUGE-L – 0.1019, USS – 0.24) while maintaining stability (cosine similarity: 0.3964). In turn, in the work of D. Martin (2025) developed CheX-LLM, which achieves high accuracy (Macro F1 – 0.9115) in labelling reports using SFT and RLHF adapted to medical narratives. Both approaches have confirmed the effectiveness of hybrid supervised learning, but the current study is more versatile due to the integration of implicit feedback, while CheX-LLM specialises in medical tasks requiring the processing of complex descriptions.

The study demonstrated that continuous online learning for LLMs reduces the loss function from 3.82

to 3.15 in 500 steps, ensuring stable adaptation without signs of overlearning. For its part, the work of S. Pratap *et al*. (2025) proposed a decomposition of LLMs to analyse the fine-tuning contribution, which reveals a decrease in Tuning Contribution in successful attacks, highlighting the impact of overfitting on the model's vulnerability. The strength of the current approach is automated adaptation to diverse feedback in real time, but the weakness is the limited number of iterations and the risk of poor-quality feedback, which limits the potential and the balance between speed and stability. At the same time, the positive aspect of the study is the accurate analysis of the impact of fine-tuning on model responses, but the weakness is the narrow focus on attacks, which limits its versatility.

The study determined that online learning for LLMs ensures model stability (cosine similarity – 0.3964) thanks to EWC, but the low Relevance metric indicates the need for improved feedback filtering. In comparison, A. Hilel *et al*. (2025) demonstrated that feedback manipulation can introduce false knowledge into LLMs, increasing vulnerability to attacks. Both approaches emphasised the importance of feedback management, but the current study adapts to a variety of real-time queries, while the above focuses on specific attack scenarios, which limits their applicability. The OpenAssistant-based GPT-2 retraining improved the quality of responses over 500 fine-tuning steps, as evidenced by the increase in ROUGE-L, USS, and stability without signs of degradation. Instead, E. Mazzullo & O. Bulut (2025) studied the retraining of GPT-3.5-turbo for automated generation of feedback to open-ended questions, which provided high user satisfaction (84.8%) and structural quality of answers (72.9%), although some of them contained linguistic inaccuracies and generalisations. Both approaches confirmed the effectiveness of GPT model retraining in improving the relevance of generation, but the current study demonstrates a higher adaptability to the flow of real online feedback, which ensures not only learning from generalised patterns but also adapting to specific changes in user behaviour.

The results obtained showed that the developed Python software implementation is suitable for productive scenarios where rapid adaptation to real user feedback is substantial with limited resources. B. Yang *et al*. (2025) proposed the Multi-Objective Repair (MORepair) method, which aims to improve the performance of LLMs in software code repair tasks through training that covers both syntactic transformations and change logic. Their results demonstrated an increase in accuracy of up to 56% but require structured data and significant computational resources. In comparison, the current system is characterised by lower computational complexity, higher flexibility to unstructured feedback, and real-time readiness, making it suitable for dynamic LLM interaction environments. This study has shown that the developed GPT-2 online learning system is effective for adapting to real feedback in conditions of limited resources and an incomplete learning cycle. Instead, the study by S.V. Gaikwad *et al*. (2024) addressed retraining of the Bidirectional Encoder Representations from Transformers and GPT-3 models to improve the accuracy and stability of emotional tone analysis. Both approaches have shown the advantages of fine-tuning in specific tasks, but the architecture proposed in the current work is better adapted to unpredictable changes in user behaviour, in contrast to tasks where the feedback structure is more homogeneous, as in sentiment analysis.

While this study demonstrated that LLMs can adapt to diverse user feedback, increasing the relevance of responses while maintaining stability, S. Karunakaran & A. Jain (2025) investigated retraining LLMs to maintain a consistent personality of AI assistants using interaction history, context, and adaptive algorithms with a focus on emotional intelligence. Both approaches emphasised the importance of adapting to user needs, but the current study stands out for its integration of implicit feedback and fast real-time adaptation, making it more versatile for dynamic environments, while another study addressed building a personalised personality that is better suited for specific scenarios, such as chatbots. The results also demonstrated that the use of RLHF in combination with SFT for continuous online learning for LLMs can be used to adapt the model to a variety of feedback, improving the relevance of answers. In turn, the study by A. Bodaghi *et al*. (2024) employed RLHF with Proximal Policy Optimiser to generate 122.951 toxic samples, which significantly improved the performance of toxicity classifiers. Both approaches have confirmed the effectiveness of RLHF for adapting LLMs, but the current study addressed universal adaptation to real-world feedback in dynamic environments, while the aforementioned ones emphasise the specific task of generating balanced data for toxicity detection.

While this study has shown that the cosine similarity and dynamics of the loss function effectively illustrates the stability and progress of the GPT-2 model adaptation. X.-K. Wu *et al*. (2025) proposed the concept of Tutorial Fine-Tuning. It integrates hermeneutical theories to further describe adaptation of LLMs, emphasising the cognitive aspects of fine-tuning to achieve a higher level of intelligence. Both approaches emphasised the importance of adapting LLMs, but the current study features a practical real-time implementation with low computational complexity, while the above focuses on theoretical determination of fine-tuning, which has less practical applicability in dynamic scenarios.

Overall, this work showed that online training of LLMs ensures adaptation to diverse user feedback with improved response quality and model stability. In comparison, K. Wang *et al*. (2025) proposed the Learning through Communication (LTC) approach, which uses a universal buffer to store linguistic and non-linguistic feedback, improving the performance of LLM agents by 3.6-12% in various tasks. Both approaches have

demonstrated the effectiveness of LLMs in adapting to dynamic conditions, but the current study is distinguished by its low computational complexity and integration of implicit feedback, while LTC focuses on optimising agents in specific environments with higher performance metrics.

The study demonstrated that the use of a hybrid RLHF approach in combination with SFT can be used to adapt GPT-2 to a variety of real-time user feedback, increasing the relevance of answers by processing both explicit and implicit feedback. Regarding the study by C. Ye *et al*. (2025), an RLHF framework with a general preference oracle that avoids the assumptions of the Bradley-Terry model and applies inverse Kullback-Leibler divergence regularisation to optimise the policy was developed, which provided an advantage over other LLMs in general preference problems. Both approaches have highlighted the flexibility of RLHF for model adaptation, but the current study is prominent for its practical implementation of continuous online learning with a focus on handling unstructured feedback in dynamic environments, while the previous study focused on theoretical versatility and algorithmic efficiency for offline and online scenarios. Compared to the study by U. Kamath *et al*. (2024), which highlighted the challenges of RLHF, the need for many human evaluations and limited scalability due to the use of multiple LLMs as reward models, the current study favoured the integration of implicit feedback for continuous online learning of LLMs. This reduces the loss function and provides a moderate improvement in quality metrics with low computational complexity. However, it is also inferior in accuracy to specialised approaches such as Constitutional AI or Direct Preference Optimisation, which solve the problem of RLHF scalability.

While the study demonstrated the ability of a continuous online learning system to adapt to diverse user feedback, increasing the relevance of answers and maintaining model stability with a limited number of iterations, the study by E. Watson *et al*. (2024) proposed a new architecture for personalising LLMs through a token-based system for addressing individual values and context. The current approach is notable for its ease of implementation and low computational complexity, which makes it more suitable for dynamic productive environments and better able to handle unstructured feedback in real time. However, the system provided deeper personalisation due to its inductive approach and user-friendly interfaces for custom annotation.

The findings of this paper highlighted that online learning for LLMs provides adaptation to diverse user feedback, improving the quality of answers and reducing the loss function, while maintaining model stability. Similarly, the study by Q. Ai *et al*. (2024) addressed the improvement of Generative Information Retrieval through continuous learning methods and processing different types of feedback, such as contextual learning in dialogues and prompt learning. However, the current approach is notable for its simplicity of implementation and ability to efficiently process unstructured feedback in real time, while the methods may be more accurate in specialised information retrieval tasks. In summary, the current study provided fast online LLMs with low computational complexity, effectively adapting to unstructured real-time feedback. Its strengths are versatility and stability, but its weakness is the limited number of iterations, which limits the quality. In contrast to specialised methods, the current approach is better suited to dynamic productive environments.

CONCLUSIONS

The developed architecture of continuous online learning for LLMs integrates a multi-level adaptation cycle based on the processing and aggregation of both explicit and implicit user feedback. The implementation of classification, weight aggregation and signal buffering can be used to form of relevant query-expected response training pairs for further hybrid retraining (SFT and RLHF) with quality control using stability, relevance and A/B testing metrics. The results confirmed that the implicit feedback significantly exceeds the explicit feedback, which determines the priority of improving the classification and weight aggregation of these signals. The proposed architecture was implemented in scalable systems with support for parallel processing, which ensures real-time updating of LLMs without loss of stability or performance, as demonstrated by the NinjaTech AI example using customised fine-tuned models.

The software implementation of the LLMs online training cycle has demonstrated effectiveness in building a scalable pipeline based on real user feedback. Two key stages have been implemented in Python: the formation of a buffer of training pairs with weighting coefficients based on the OpenAssistant open data set and further training of the GPT-2 model using the Transformers library. In the process of data preparation, 21,265 query-expected-response examples were generated, incorporating relevance and length filters, and weights were determined based on user interaction. Fine-tuning GPT-2 on 52,912 examples with a limit of 500 steps ensured a steady decrease in the loss function from 3.82 to 3.15, without any signs of degradation or instability. The results showed that the model can effectively adapt even with partial retraining, demonstrating the advantages of the online approach over static and offline models.

A comprehensive evaluation of the results of adaptive re-training of LLMs showed a moderate but stable improvement in the quality of GPT-2 model answers after 500 fine-tuning steps on real user feedback. The automatic metrics (ROUGE-L = 0.1019, BLEU = 0.0060) showed an increase in lexical and semantic correspondence with the expected answers, and the UX metrics (USS = 0.1883) indicated the potential for further optimisation through better feedback sampling. The dynamics of learning loss confirmed the effectiveness

of the pipeline, and the cosine similarity analysis (mean value 0.3964) showed that the basic knowledge was retained with partial changes in answers, which indicates a balance between adaptation and stability. Examples of transformations of answers confirmed the growth of content, specificity and personalisation.

However, the study has certain limitations, such as the limited amount of feedback, the use of only one model architecture (GPT-2), and the reduced number of retraining steps. This reduces the possibility of generalising the results to more modern LLMs and larger application scenarios. Further research should focus on testing the approach at LLMs, expanding the feedback base, and studying the impact of different types of feedback on the quality of adaptation in the long term.

ACKNOWLEDGEMENTS

None.

FUNDING

None.

CONFLICT OF INTEREST

None.

REFERENCES

[1] Ai, Q., Dou, Z., & Zhang, M. (2024). Improving generative information retrieval systems based on user feedback. In R.W. White & C. Shah (Eds.), *Information access in the era of generative AI* (pp. 111-133). Cham: Springer. doi: 10.1007/978-3-031-73147-1_5.

[2] Anisuzzaman, D.M., Malins, J.G., Friedman, P.A., & Attia, Z.I. (2025). Fine-tuning LLMs for specialized use cases. *Mayo Clinic Proceedings Digital Health*, 3(1), article number 100184. doi: 10.1016/j.mcpdig.2024.11.005.

[3] Balaskas, G., Papadopoulos, H., Pappa, D., Loisel, Q., & Chastin, S. (2025). A framework for domain-specific dataset creation and adaptation of large language models. *Computers*, 14(5), article number 172. doi: 10.3390/computers14050172.

[4] Bodaghi, A., Fung, B.C.M., & Schmitt, K.A. (2024). AugmenToxic: Leveraging reinforcement learning to optimize LLM instruction fine-tuning for data augmentation to enhance toxicity detection. *ACM Transactions on the Web*. doi: 10.1145/3700791.

[5] Cardó, A.V. (2025). *Investigating feedback types in reinforcement learning with human feedback and large language models*. (Bachelor's thesis, Aalto University, Espoo, Finland).

[6] Dombi, J., & Jónás, T. (2022). Weighted aggregation systems and an expectation level-based weighting and scoring procedure. *European Journal of Operational Research*, 299(2), 580-588. doi: 10.62441/nano-ntp.vi.5447.

[7] Gaikwad, S.V., Agarkar, P., Mohapatra, S., & Bagade, S. (2024). Fine-tuning LLM for sentiment analysis. *Nanotechnology Perceptions*, 20(6), 4946-4959. doi: 10.62441/nano-ntp.vi.5447.

[8] Haider, Z., Rahman, H., Devabhaktuni, V., Moeykens, S., & Chakraborty, P. (2025). A framework for mitigating malicious RLHF feedback in LLM training using consensus-based reward. *Scientific Reports*, 15, article number 9177. doi: 10.1038/s41598-025-92889-7.

[9] Hao, S., & Duan, L. (2025). Online learning from strategic human feedback in LLM fine-tuning. In *CASSP 2025 – 2025 IEEE international conference on acoustics, speech and signal processing* (pp. 1-5). Hyderabad: IEEE. doi: 10.1109/ICASSP49660.2025.10887891.

[10] Hilel, A., Shenfeld, I., Andreas, J., & Choshen, L. (2025). LLM hypnosis: Exploiting user feedback for unauthorized knowledge injection to all users. *ArXiv*. doi: 10.48550/arXiv.2507.02850.

[11] Kamath, U., Keenan, K., Somers, G., & Sorenson, S. (2024). Tuning for LLM alignment. In *Large language models: A deep dive* (pp. 177-218). Cham: Springer. doi: 10.1007/978-3-031-65647-7_5.

[12] Kampelopoulos, D., Tsanousa, A., Vrochidis, S., & Kompatsiaris, I. (2025). A review of LLMs and their applications in the architecture, engineering and construction industry. *Artificial Intelligence Review*, 58, article number 250. doi: 10.1007/s10462-025-11241-7.

[13] Karunakaran, S., & Jain, A. (2025). Fine-tuning LLMs for personality preservation in AI assistants. *International Journal of Research in Modern Engineering & Emerging Technology*, 13(4), 172-191. doi: 10.63345/ijrmeet.org.v13.i4.10.

[14] Lin, X., Wang, W., Li, Y., Yang, S., Feng, F., Wei, Y., & Chua, T. (2024). Data-efficient fine-tuning for LLM-based recommendation. In *Proceedings of the 45th international ACM SIGIR conference on research and development in information retrieval* (pp. 365-374). New York: Association for Computing Machinery. doi: 10.1145/3626772.3657807.

[15] Martin, D. (2025). Improving CXR report labeling through LLM fine-tuning and human feedback. *Preprints*. doi: 10.20944/preprints202504.1668.v1.

[16] Mazzullo, E., & Bulut, O. (2025). Automated feedback generation for open-ended questions: Insights from fine-tuned LLMs. In *Proceedings of large foundation models for educational assessment* (pp. 103-120). Vancouver: MLResearchPress.

[17] Next Electronics. (n.d.). *Elastic weight consolidation (EWC)*. Retrieved from https://www.next.gr/ai/deep-learning-theory/elastic-weight-consolidation-ewc/.

[18] NinjaTech AI. (n.d.). *Ninja LLM suite*. Retrieved from https://www.ninjatech.ai/product/ninja-llm.
[19] OpenAssistant conversations dataset. (n.d.). Retrieved from https://huggingface.co/datasets/OpenAssistant/oasst1.
[20] Pratap, S., Aranha, A.R., Kumar, D., Malhotra, G., Iyer, A.P.N., & Shylaja, S.S. (2025). The fine art of fine-tuning: A structured review of advanced LLM fine-tuning techniques. *Natural Language Processing Journal*, 11, article number 100144. doi: 10.1016/j.nlp.2025.100144.
[21] Punnaivanam, M., & Velvizhy, P. (2024). Contextual fine-tuning of language models with classifier-driven content moderation for text generation. *Entropy*, 26(12), article number 1114. doi: 10.3390/e26121114.
[22] Rawal, N., Tavva, P., & Selvakumar, P. (2024). Enhancing large language model performance with reinforcement learning from human feedback: A comprehensive study on Q&A, summarization, and classification. In *2024 international conference on electrical, computer and energy technologies* (pp. 1-6). Sydney: IEEE. doi: 10.1109/ICECET61485.2024.10698396.
[23] Rehan, S., Al-Bander, B., & Al-Said Ahmad, A. (2025). Harnessing large language models for automated software testing: A leap towards scalable test case generation. *Electronics*, 14(7), article number 1463. doi: 10.3390/electronics14071463.
[24] Shi, T., *et al*. (2024). WildFeedback: Aligning LLMs with *in-situ* user interactions and feedback. *ArXiv*. doi: 10.48550/arXiv.2408.15549.
[25] Terras, N., Pereira, F., Silva, A.R., Santos, AA., Lopes, A.M., Silva, A.F.D., Cartal, L.A., Apostolescu, T.C., Badea, F., & Machado, J. (2025). Integration of deep learning vision systems in collaborative robotics for real-time applications. *Applied Sciences*, 15(3), article number 1336. doi: 10.3390/app15031336.
[26] Wang, J., Lu, H., Liu, Y., Ma, H., Wang, Y., Gu, Y., Zhang, S., Han, N., Bi, S., Baugher, L., Chi, E.H., & Chen, M. (2024). LLMs for user interest exploration in large-scale recommendation systems. In T. di Noia, P. Lops, T. Joachims. K. Verbert, P. Castells, Z. Dong & B. London (Eds.), *Proceedings of the 18th ACM conference on recommender systems* (pp. 872-877). New York: Assoviation Computing Machinery. doi: 10.1145/3640457.3688161.
[27] Wang, K., Lu, Y., Santacroce, M., Gong, Y., Zhang, C., & Shen, Y. (2025). Adapting LLM agents with universal communication feedback. In *Findings of the association for computational linguistics* (pp. 6105-6122). Albuquerque: Association for Computational Linguistics. doi: 10.18653/v1/2025.findings-naacl.339.
[28] Watson, E., Viana, T., Zhang, S., Sturgeon, B., & Petersson, L. (2024). Towards an end-to-end personal fine-tuning framework for AI value alignment. *Electronics*, 13(20), article number 4044. doi: 10.3390/electronics13204044.
[29] Wu, X.-K., *et al*. (2025). LLM fine-tuning: Concepts, opportunities, and challenges. *Big Data and Cognitive Computing*, 9(4), article number 87. doi: 10.3390/bdcc9040087.
[30] Yang, B., Tian, H., Ren, J., Zhang, H., Klein, J., Bissyandé, T.F., Le Goues, C., & Jin, S. (2025). MORepair: Teaching LLMs to repair code via multi-objective fine-tuning. *ACM Transactions on Software Engineering and Methodology*. doi: 10.1145/3735129.
[31] Ye, C., Xiong, W., Zhang, Y., Dong, H., Jiang, N., & Zhang, T. (2025). Online iterative reinforcement learning from human feedback with general preference model. In *Proceedings of the 38th international conference on neural information processing systems* (pp. 81773-81807). Red Hook: Curran Associates Inc.
[32] Zhao, Q., Harper, F.M., Adomavicius, G., & Konstan, J.A. (2018). Explicit or implicit feedback? Engagement or satisfaction? A field experiment on machine-learning-based recommender systems. In *Proceedings of the 33rd annual ACM symposium on applied computing* (pp. 1331-1340). New York: Association for Computing Machinery. doi: 10.1145/3167132.3167275.

Безперервні цикли зворотного зв'язку: онлайн-налаштування LLM за допомогою сигналів користувача

Софія Швець

Магістр, спеціаліст з обробки даних
NinjaTech AI
94022, вул. Ель Каміно Реал, 4410, м. Лос-Алтос, Сполучені Штати Америки https://orcid.org/0009-0000-7896-6479

Анотація. Інтенсивне зростання використання мовних моделей реального часу вимагає механізмів їх динамічної адаптації до змін у запитах, термінології та очікуваннях користувачів. Метою дослідження було вивчення підходів до перенавчання великих мовних моделей на основі безперервного зворотного зв'язку. Для досягнення цієї мети було застосовано теоретичне та структурно-функціональне моделювання архітектури адаптації, експериментальну реалізацію циклу перенавчання мовної моделі з обробкою та класифікацією різних типів зворотного зв'язку, а також кількісну оцінку результатів за допомогою автоматичних та користувацьких метрик. Результати дослідження показали ефективність архітектури безперервного онлайн-навчання, яка забезпечує актуальність та стабільність мовної моделі в реальному часі. У дослідженні визначено, що неявний зворотний зв'язок зустрічається в 4-10 разів частіше, ніж явний зворотний зв'язок, але явний зворотний зв'язок дає вищий приріст точності відповідей. Запропонована система успішно інтегрує різні типи користувацьких сигналів, забезпечуючи динамічну генерацію навчальних прикладів та гібридне перенавчання, зберігаючи при цьому якість та узгодженість результатів. Програмний цикл Python для адаптивного перенавчання мовної моделі включав обробку та фільтрацію користувацьких сигналів для формування високоякісного буфера навчальних пар. Після 500 кроків перенавчання на 52 912 парах запит-відповідь спостерігалося значне покращення моделі, що підтверджувалося зменшенням функції втрат з 3,82 до 3,15 та стабільністю процесу точного налаштування без ознак перенавчання. Результати попереднього навчання показали помірне покращення якості відповідей після адаптації: лексична подібність за даними Recall-Oriented Understudy for Gisting Evaluation становила 0,102, точність за даними Bilingual Evaluation Understudy – 0,006, а суб'єктивна задоволеність користувачів зросла до 0,24, зберігаючи при цьому стабільність моделі із середнім значенням косинусної подібності 0,396. Підхід, запропонований у цьому дослідженні, покращує якість та релевантність відповідей мовних моделей у реальному часі, зберігаючи їх стабільність, і може бути використаний у продуктивних системах для покращення користувацького досвіду

Ключові слова: адаптивне перенавчання; генеративні трансформатори; адаптація динамічної моделі; реалізація на Python; гібридне навчання; метрики оцінки якості; стабільність мовної моделі