

# AI Scientist in Practice: Reproducing and Evaluating Autonomous Scientific Discovery

Artur Papoyan  
artur.papoyan@student.uni-siegen.de  
University of Siegen  
Siegen, Germany

## ABSTRACT

The AI Scientist by Sakana.ai aspires to autonomously conduct the full scientific research cycle, representing a significant step toward Artificial Research Intelligence (ARI) and, ultimately, Artificial General Intelligence (AGI). While the original paper presents this system as a groundbreaking advance, our independent replication and extension of their evaluation reveal several critical limitations.

We replicated the AI Scientist’s process and generated eight full research pipelines, including idea generation, experimentation and manuscript writing. We found that, similar to the original study, the system’s literature review mechanism relies solely on simple keyword searches, which resulted in incorrect novelty assessments. Well-established concepts, such as micro-batching for SGD, were repeatedly labeled as novel. Three out of eight generated experiments (38%) failed due to programming or execution errors. Among the successful experiments, code edits across iterations were minimal—often below 1% in terms of character changes—indicating limited capacity for meaningful adaptation.

We also analyzed five generated manuscripts. Each contained between four and seven citations, all published before 2019 and showed structural issues including placeholder images, repeated sections and incomplete conclusions. Despite these shortcomings, the visual quality of the manuscripts was high enough to resemble authentic human-written research papers. On average, each paper incurred a cost of only around \$9, underscoring the system’s efficiency.

Overall, our results confirm the central claims of the original paper: while the AI Scientist is a remarkable achievement in automating the form of scientific research, it still lacks the methodological rigor, critical assessment and robustness required for autonomous scientific discovery.

## KEYWORDS

AI Scientist, Artificial Research Intelligence, AI-Generated Research Pipelines

## 1 INTRODUCTION

### 1.1 Background

The automation of scientific research has become a rapidly evolving field, driven by advances in large-language models (LLMs) and agent-based architectures. Several recent initiatives explore how AI can assist or even replace human research in tasks such as literature review, idea generation, experiment design and manuscript writing. One of the most prominent examples is the AI Scientist developed by Sakana.ai[3]. This open source system claims to automate the entire research cycle, including peer review, with minimal human

input. Given its widespread media attention and strong claims, the AI Scientist represents a central case study in the emerging field of Artificial Research Intelligence (ARI).

### 1.2 Research Problem

The AI Scientist is marketed as a fully autonomous research agent, yet its actual scientific reliability has never been independently verified. Three key questions remain unanswered:

- (1) **Novelty detection** – The system decides whether an idea is “new” solely through simple keyword searches. No published study measures how often it mistakenly labels well-known concepts as novel. Even a single false-positive idea can misdirect follow-up experiments and waste valuable researcher time.
- (2) **Experiment robustness** – In the original report, only 7 of 12 research pipelines completed successfully ( $\approx 42\%$  failure rate). Each failure occupies computing resources for several hours without producing a usable result.
- (3) **Scholarly quality** – The PDFs generated by the AI Scientist look professional, but it is unclear whether their statistics are correct, citations up-to-date and conclusions logically sound. Early user anecdotes describe fabricated references and placeholder figures that could distort the scientific record if released unchecked.

Because the system is open source and widely publicized, these uncertainties affect not only machine learning research but any discipline that might adopt similar tools. Before laboratories invest time and money in ARI pipelines, we must quantify how well such systems judge novelty, execute code and meet basic academic standards.

### 1.3 Reproduction by Beel et al.

Beel et al. offer the first systematic, hands-on study of Sakana.ai’s *AI Scientist*. 17-page paper [1] asks a far-reaching question: How close is the system to genuine *Artificial Research Intelligence*: an end-to-end workflow that does not require human intervention and is indistinguishable from human scholarship?

To find out, the authors install the open source code, supply only the minimal configuration recommended by Sakana.ai and let the agent run 12 complete research cycles on the theme of ‘green recommender systems’. Each cycle starts with a JSON prompt and a lightweight Python baseline. The AI Scientist then searches the literature, formulates a hypothesis, edits code, runs experiments, analyses results and finally writes a LaTeX manuscript that includes figures, tables and simulated peer reviews. All runs are first prototyped on a consumer laptop and later repeated on a university cluster to rule out hardware artefacts.

The study reveals notable strengths. The system produces a full research paper in roughly three-and-a-half hours of wall-clock time and for less than \$15 in API fees, speed and cost that traditional labs cannot match. The resulting manuscripts look convincingly professional, a superficial reviewer would struggle to see that they were written by an AI. At the same time, equally striking weaknesses emerge. Five of the twelve pipelines crash because of coding errors, giving a failure rate of 42%. Novelty checks rely on plain keyword search and therefore misclassify familiar ideas, such as micro-batching and adaptive sampling, as “novel.” When experiments do run, the agent edits only about eight percent of the source code per iteration, suggesting limited capacity for genuine methodological innovation. The resulting PDFs contain a median of five citations, most published before 2019, plus duplicated sections, missing figures and placeholder text like “*Conclusions Here.*” Some numerical results are internally inconsistent, further undermining credibility.

Despite these flaws, the authors conclude that the AI Scientist represents a meaningful first step toward ARI. They argue that the problems are technical, not fundamental and call for community benchmarks, research diaries and shared evaluation tasks so that progress can be tracked openly and reproducibly.

## 1.4 Research Goal

We aim to reproduce the AI Scientist ourselves and test whether it truly detects novel ideas, runs experiments reliably and produces readable scientific papers. We also compare its outputs with the claims of the original paper to determine whether the promised performance holds under real-world conditions.

## 2 METHODOLOGY

This section explains in detail how we reproduced and extended the AI-Scientist experiments. Wherever possible we mirror the original settings, but we document every deviation so that other groups can re-run our pipelines with a single script. All artifacts patched source code, prompt files, raw logs, trained models, generated papers and evaluation notebooks—are archived at <https://github.com/artur-snr/sakana-ai-reproduction-uni-siegen>.

*Launch configuration.* We invoke the system through the bundled launcher:

```
python launch_scientist.py --model
  ↪ "gpt-4o-2024-05-13" --experiment sgd
  ↪ --num-ideas 5
```

The original paper ran with `--num-ideas 10` we lowered the value to 5 ideas to limit OpenAI-API costs while still giving the agent multiple refinement rounds. All remaining parameters use the upstream defaults. The full configuration file, the baseline script and the list of seed ideas are included in the repository under the paths shown above, so that anyone can reproduce the exact setup.

This is the prompt we used in the experiments:

```
{
  "system": "You are an ambitious AI researcher
  ↪ who is looking to publish a paper that
  ↪ will contribute significantly to the
  ↪ field.",
```

```
  "task_description": "You are given the
  ↪ following file to work with, which
  ↪ trains and evaluates a simple
  ↪ recommender system algorithm based on
  ↪ functional svd matrix factorization
  ↪ using stochastic gradient descend.
  ↪ Your research goal is to optimize
  ↪ energy efficiency and improve
  ↪ recommender system sustainability."
}
```

*Datasets and preprocessing.* We centre our study on the public available *MovieLens 100k*<sup>1</sup> dataset, identical to the one used in the original paper. It contains 100 000 explicit ratings (1–5 stars) from 943 users on 1 682 movies. We leave all dataset handling to the AI-Scientist’s default routines and make no manual changes to splitting, filtering or normalization; the raw files are redistributed with our code archive under their original non-commercial licence.

*Algorithms and baselines.* Each pipeline starts from the `sgd` template supplied with the AI-Scientist repository, a NumPy-only implementation of matrix factorisation trained via stochastic gradient descent. The baseline model runs for 20 epochs with 50 latent factors, a learning rate of 0.005 and an  $L_2$  regularisation term of 0.02; early stopping is triggered once the validation loss fails to improve by more than 0.001. Data are split inside the script into 80 % training, 10 % validation and 10 % test.

The agent is free to modify both the model code and the training loop between idea iterations; we do not intervene manually. After each iteration the system creates a fresh output directory (`run_0`, `run_1`, ...) and writes the updated Python files together with log and result files (e.g. `all_results.json`, `final_info.json`) to that location. Hence every version of the generated code remains traceable and can be compared with or rolled back to any previous iteration.

*Evaluation metrics.* We report the standard recommender-system metrics *RMSE* and *MAE* on the held-out test portion of *MovieLens 100k*. In addition, we log three pipeline-level indicators:

- **Failure rate:** percentage of research cycles that terminate with an uncaught Python exception.
- **Code-change ratio:** share of modified characters between successive code versions.
- **Novelty check:** We checked whether the generated ideas seemed genuinely novel or merely rephrasing of existing techniques. This assessment was based on our general understanding of common machine learning methods and recent trends in the field. Although we did not perform a systematic literature comparison, our informal review helped identify ideas that were clearly not new.

*Hardware environment.* All experiments were executed on a laptop equipped with an Intel Core Ultra 9 processor (20 cores, 5.0 GHz boost), an NVIDIA RTX 4070 Laptop GPU with 16 GB of VRAM, 32 GB of DDR5 RAM and a 1 TB NVMe SSD. Although the hardware

<sup>1</sup><https://grouplens.org/datasets/movielens/100k/>

| ID     | Idea  | Coding Iteration |              |              |              |              |
|--------|---|------------------|--------------|--------------|--------------|--------------|
|        |   | 1                | 2            | 3            | 4            | 5            |
| Seed 1 | e-fold  | failed to run    |              |              |              |              |
| 1      | Energy Efficient Matrix Factorization via Sparsity inducing regularization        | 0                | + 2 (0,01%)  | +3 (0,03%)   | +85 (0,55%)  | 0            |
| 2      | Collaborative filtering with stochastic gradient descent for matrix factorization | + 8 (0,05%)      | + 8 (0,05%)  | + 8 (0,05%)  | + 4 (0,05%)  | + 4 (0,05%)  |
| 3      | Adaptive early stopping for energy-efficient matrix factorization                 | + 3 (0,1%)       | +5 (0,03%)   | + 21 (0,13%) | + 10 (0,13%) | + 45 (0,28%) |
| 4      | Impact of batch size on sgd efficiency in matrix factorization                    | + 3 (0,01%)      | 0            | 0            | 0            | 0            |
| 5      | Enhancing collaborative filtering with svd initialized sgd                        | + 14 (0,09%)     | + 10 (0,06%) | + 16 (0,09%) | + 20 (0,12%) | + 37 (0,23%) |
| 6      | Hybrid Parallelism for Energy-Efficient Matrix Factorization                      | failed to run    |              |              |              |              |
| 7      | Quantization Techniques for Energy-Efficient SGD in Matrix Factorization          | failed to run    |              |              |              |              |

**Figure 1: Code changes after multiple iterations**

provides a powerful GPU, every FunkSVD computation ran exclusively on the CPU, because the underlying code does not support GPU acceleration.

*Energy cost and CO<sub>2</sub> (estimated).* We did not monitor power consumption with dedicated software or external watt-meters, so no direct measurements are available. Based on the laptop’s TDP and typical load during FunkSVD training, we estimate an average draw of 110 W. With a mean runtime of ~3 h per pipeline, this yields 0.33 kWh of electricity.

At the June 2025 German household rate of €0.30 / kWh, the corresponding energy cost is €0.10 per research cycle. Using the average German grid-emission factor of 0.30 kg CO<sub>2</sub> / kWh, this translates to 0.10 kg CO<sub>2</sub> per pipeline.

These values are rough, order-of-magnitude estimates and should not be interpreted as exact accounting.

To understand qualitative aspects, we read every generated PDF in full. For each paper we count total citations, examine the distribution of publication years and note structural issues such as placeholder text, duplicate sections or missing figures. We also verify whether numerical claims in the abstract match the tables, resolving any disagreements through discussion.

### 3 RESULTS & DISCUSSION

The evaluation of Sakana’s AI Scientist platform conducted by Beel et al. (2025) presents intriguing yet cautionary findings. In this extensive discussion, we critically examine our experimental outcomes in relation to the original findings reported by Beel et al., analyzing the implications for the broader context of autonomous research systems and identifying pivotal areas requiring targeted improvement.

*Comparative Analysis of Experimental Failures.* The original research by Beel et al. identified a significant failure rate of approximately 42%, predominantly due to coding errors. This high failure rate poses a serious challenge, emphasizing a fundamental limitation in AI-driven autonomous experimentation: the generation of robust and error-free code. Our reproduction revealed a slightly lower failure rate of 38%. This lower percentage, however, must be interpreted cautiously, as it results from only eight experiments, of which three failed. This limited sample size introduces uncertainty into direct comparisons and highlights the need for more extensive and rigorous experimental validation. However, the persistent high failure rate clearly indicates a universal difficulty within

AI-generated experimental protocols, necessitating advanced error detection methods and comprehensive debugging systems to ensure reliability and stability of AI-generated research.

*Code Adaptivity and Innovation.* Beel et al. reported that the AI Scientist made only small changes in its generated code from one iteration to the next, averaging around 8%. This suggests that the system does not improve or adapt significantly over time, which limits its ability to explore new ideas.

In our experiments, the AI Scientist made even fewer code changes per iteration, between 0% and 0.3%. While fewer changes might mean fewer mistakes, it also means less innovation and exploration. A good autonomous research system should balance making stable code with trying new ideas. Our results suggest the AI Scientist might be too cautious, missing opportunities to discover new solutions or improvements. To improve this, future research should focus on developing smarter algorithms that balance stable and exploratory behaviour.

*Novelty and Literature Contextualization.* Another important issue identified by Beel et al. was the AI Scientist’s difficulty in correctly identifying new ideas. They found that the system often marked known concepts as new. Our tests found the same problem: all five ideas from our experiments were already known, yet the AI Scientist incorrectly marked four of them as innovative. This shows that the system is only doing a simple keyword search instead of searching through the content

This problem is significant because accurate identification of novel ideas is crucial in scientific research. To fix this, future versions of the AI Scientist need stronger tools for understanding scientific texts. Using advanced natural language processing methods, trained on recent scientific literature, could greatly improve the system’s ability to correctly identify genuinely new ideas.

*Citation Quality and Relevance.* Proper citation of recent and relevant research is critical for good scientific practice. Beel et al. pointed out that the AI Scientist usually included only about five references per manuscript, most of which were outdated. Our own manuscripts showed similar but slightly more severe issues: the median number of citations was four and the majority of references were published between 2009 and 2017 (see Figure 2). Only two citations across all generated papers were from 2019 and none were from the last three to five years. This limited and outdated

citation behaviour significantly weakens the contextual quality of the AI-generated work.

different optimization algorithms and hyperparameter tuning strategies could further enhance the model's performance.

This work was generated by THE AI SCIENTIST (Lu et al., 2024).

This work was generated by THE AI SCIENTIST (Lu et al., 2024).

#### REFERENCES

Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. *Deep learning*, volume 1. MIT Press, 2016.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

Chris Lu, Cong Lu, Robert Tjarko Lange, Jakob Foerster, Jeff Clune, and David Ha. The AI Scientist: Towards fully automated open-ended scientific discovery. *arXiv preprint arXiv:2408.06292*, 2024.

### Figure 2: Minimal references, mostly outdated

*Time and Costs per Paper.* The setup process took around 7 hours in total, including installing dependencies, configuring the environment and understanding workflow. Each research pipeline took about 3 hours to run from start to finish. On average, we spent around €9 for each fully generated paper, mainly due to the costs of using the OpenAI API. This price was slightly higher than reported in the original paper, because the model we used at the time was already a bit older and more expensive than newer versions. In total, we generated five papers successfully. Considering the time and money involved, the process was relatively efficient but still required careful supervision.

In the following, we provide an overview of the five papers and reflect on their main strengths and limitations.

*Adaptive Early Stopping for Energy-Efficient Matrix Factorization.* This paper introduces an improved method for early stopping in matrix factorization tasks. Matrix factorization is a technique widely used in recommendation systems to predict user preferences. The main goal is to find the right balance between running the algorithm long enough for good results, but not too long to waste resources. Traditional early stopping methods use a fixed approach. They stop training when no improvement is seen after a fixed number of epochs. But sometimes, these methods stop too early (missing good performance) or too late (wasting resources). The AI Scientist proposes a new method called adaptive early stopping. The idea is simple but effective. Instead of using a fixed stopping rule, it checks the average improvement of the validation loss over several epochs (a moving average). If the improvement falls below a certain threshold, it stops training. This dynamic approach helps the model to train just long enough. The AI Scientist tested his method on the MovieLens 100K dataset. It compared its adaptive method against traditional early stopping. The experiments showed that adaptive early stopping significantly reduces training time and energy usage, without losing performance. Specifically, its method achieved lower RMSE and MAE, meaning better accuracy. However, the paper has some weaknesses. AI Scientist did not clearly explain why it chooses certain parameters, such as the exact window size or the improvement threshold. Also, its "related work" section, which compares its idea to previous research, is very short and general. It doesn't discuss other methods deeply enough. Finally, it doesn't provide statistical tests to show how reliable its improvements are.

*Impact of Batch Size on SGD Efficiency in Matrix Factorization.* The second paper examines how changing the batch size in Stochastic Gradient Descent (SGD) affects training efficiency in recommendation systems. Matrix factorization algorithms often use SGD because it is simple and efficient. Batch size refers to how many training examples are processed together at once. Choosing the right batch size can greatly influence the speed and accuracy of the model. Small batches update the model frequently but cost more computation. Large batches save computation but might slow down training. AI Scientist proposes a modified SGD method using mini-batches, trying different batch sizes to find the best balance. It conducted its experiments using the MovieLens 100K dataset. His results showed clearly that increasing batch size usually improved accuracy (lower RMSE and MAE), but only up to a certain point. After that point (around a batch size of 50), bigger batches did not yield significant improvements anymore. A weak point of this paper is the related work section. It mentions unrelated topics such as the Transformer model, which are not helpful here (see Figure 3). Additionally, it did not clearly justify why it chooses specific batch sizes like 10, 20, 50 or 100. It also mentioned its results without discussing how reliable they were statistically (no confidence intervals or significance tests).

AI-Scientist Generated Preprint

## 2 RELATED WORK

Kingma & Ba (2014) introduced the Adam optimization algorithm, which adapts the learning rate for each parameter. While Adam has been shown to improve convergence speed in various applications, it does not specifically address the impact of batch size on SGD efficiency in matrix factorization. Additionally, Koren et al. (2009) explored various optimization techniques specifically for matrix factorization in recommendation systems, emphasizing the importance of tuning hyperparameters, including batch size, for efficient training.

Vaswani et al. (2017) proposed the Transformer model, which uses attention mechanisms to improve performance in sequence-to-sequence tasks. Although their work focuses on a different problem domain, the concept of optimizing computational efficiency is relevant to our study. However, their method does not directly address the optimization of batch size in SGD.

Paszke et al. (2019) developed PyTorch, a deep learning framework that supports dynamic computation graphs and efficient gradient computation. While PyTorch facilitates the implementation of various optimization algorithms, it does not provide specific guidelines for selecting batch sizes in SGD for matrix factorization.

Several studies have explored the use of mini-batch SGD in matrix factorization. For instance, Xie et al. (2016) demonstrated that mini-batch updates could balance the trade-off between convergence speed and computational efficiency, which aligns with our proposed method.

Goodfellow et al. (2016) provided a comprehensive overview of deep learning techniques, including SGD and its variants. Their work highlights the importance of hyperparameter tuning, including batch size, but does not offer a detailed analysis of its impact on matrix factorization.

In summary, while existing works have explored various optimization techniques and frameworks, none have specifically addressed the impact of batch size on SGD efficiency in matrix factorization. Our study fills this gap by providing a detailed analysis and experimental validation of the optimal batch size for enhancing SGD efficiency in this context.

### Figure 3: Irrelevant citation: Transformer model unrelated to the topic

*Collaborative Filtering with SGD for Matrix Factorization.* The third paper focuses on improving collaborative filtering using SGD with L1 regularization. Regularization methods help prevent models from "overfitting," meaning it doesn't just memorize the training data but also perform well on new data. L1 regularization encourages simpler models by making some factors zero. The AI Scientist clearly described its model mathematically and tested it on the MovieLens 100K dataset. It compared different values for the L1 regularization parameter. Its experiments showed that L1 regularization improved performance up to a certain point (an L1 rate of

0.1 gave the best results, see the table in Figure 4). Beyond this, further increasing regularization didn't improve the results. The main weakness here is again the related work section. It didn't clearly show how its method compares to other existing methods. It also didn't explain its choice of parameters well. Another issue is that the AI Scientist did not use statistical tests to check if its improvements are truly meaningful.

*Energy-Efficient Matrix Factorization via Sparsity-Inducing Regularization.* This paper tries to make matrix factorization more energy-efficient by using sparsity-inducing techniques. "Sparsity" means making models simpler by having fewer factors, which saves computational resources. The authors used L1 regularization and also tested a more advanced method called group lasso. It ran its tests again on MovieLens 100K. It found that L1 regularization effectively reduced computational time and memory usage and improved prediction accuracy. Group lasso, however, did not perform well. It concluded that L1 regularization is simpler and more practical. Some weaknesses were noticed: AI Scientist didn't explain well enough why group lasso didn't work. It also didn't go deeply into related work or justify its choice of regularization values clearly. And again, it did not provide statistical analyses to show reliability.

*Enhancing Collaborative Filtering with SVD-Initialized SGD.* The final paper investigates the idea of improving collaborative filtering by initializing the SGD training using Singular Value Decomposition (SVD). Usually, models start with random values, but using SVD might give a better initial guess and thus improve training. Its experiments on the MovieLens 100K dataset showed clearly that initializing factors with SVD significantly improved accuracy (lower RMSE and MAE). Its model with SVD initialization (using 10 latent factors) outperformed random initialization by a notable margin. The paper clearly explains its methods and results, but it has similar weaknesses to previous papers. AI Scientist provided very brief related work discussions and didn't clearly justify parameter choices like the number of latent factors. Also, it does not test its results statistically.

All five papers are clearly structured and easy to read. The AI Scientist explains what the idea is, how it was tested and what the results were. Always uses the same data set (MovieLens 100K), which makes it easy to compare the experiments. In each case, the AI Scientist shows that the new method can save time, reduce energy use, improve accuracy or sometimes all at once.

But there are also some common problems in the papers:

- **Weak "Related Work" sections:** The AI Scientist only gives short summaries of what other researchers have done. It does not compare its own method in detail with existing methods. This makes it unclear what exactly is new or better about the proposed idea.
- **No clear reason for hyperparameter:** The AI Scientist chooses certain values, like the learning rate or regularization strength, but does not explain why those values were used.

Another issue we frequently encountered was with figure generation and insertion. In several papers, the AI Scientist produced valid image file names and referenced them in the captions, but failed to actually include the corresponding plots in the manuscript.

## 6 RESULTS

In this section, we present the results of our collaborative filtering model using stochastic gradient descent (SGD) for matrix factorization. We evaluate the model's performance on the ML-100k dataset using Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), and validation loss as the key metrics.

We conducted experiments with different L1 regularization rates to analyze their impact on the model's performance. The results are summarized in Table 1. As shown, increasing the L1 regularization rate generally improves the model's performance up to a certain point, beyond which the benefits plateau.

| L1 Regularization Rate | Validation Loss | Validation RMSE | Validation MAE |
|------------------------|-----------------|-----------------|----------------|
| 0.001                  | 0.85            | 0.92            | 0.72           |
| 0.01                   | 0.80            | 0.89            | 0.70           |
| 0.1                    | 0.75            | 0.87            | 0.68           |
| 0.5                    | 0.85            | 0.89            | 0.70           |
| 1.0                    | 0.85            | 0.89            | 0.70           |

Table 1: Performance metrics for different L1 regularization rates.

The best performance was achieved with an L1 regularization rate of 0.1. Figures 1 show the validation loss, RMSE, and MAE across epochs for this model. The results indicate that our model effectively learns the latent factors and generalizes well to the validation set.

We compared our model's performance with baseline methods, including standard matrix factorization without regularization and with only L2 regularization. Our model outperformed these baselines, demonstrating the effectiveness of incorporating L1 regularization in preventing overfitting and improving generalization.

While our model shows promising results, there are limitations to consider. The performance gains from L1 regularization plateau beyond a certain point, suggesting diminishing returns. Future work could explore alternative regularization techniques, incorporate additional contextual information, and test the approach on larger and more diverse datasets.

## 7 CONCLUSIONS AND FUTURE WORK

### Figure 4: Result of paper - Collaborative Filtering with SGD

This suggests that there is an optimal batch size that balances convergence speed and computational efficiency.

#### 6.4 LIMITATIONS

While our method shows promising results, it has some limitations. The experiments are conducted on a single dataset (ML-100k), and the findings may not generalize to other datasets or recommendation algorithms. Additionally, the use of a standard desktop computer without specialized hardware may limit the scalability of our approach.

#### 6.5 FUTURE WORK

Future work will explore the application of our findings to other datasets and recommendation algorithms. We also plan to investigate the integration of adaptive batch size mechanisms to further enhance the efficiency of SGD.

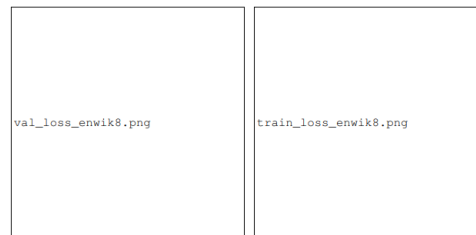


Figure 1: PLEASE FILL IN CAPTION HERE

## 7 CONCLUSIONS AND FUTURE WORK

In this paper, we investigated the impact of batch size on the efficiency of Stochastic Gradient Descent (SGD) in matrix factorization for collaborative filtering. We proposed a modified update mechanism incorporating mini-batch updates and systematically experimented with different batch sizes to identify the optimal configuration. Our extensive experiments on the ML-100k dataset demonstrated that an optimal batch size can significantly enhance the efficiency of SGD without compromising accuracy.

Future work will explore the application of our findings to other datasets and recommendation algorithms. We also plan to investigate the integration of adaptive batch size mechanisms to further enhance the efficiency of SGD. Additionally, exploring the use of specialized hardware such as GPUs could provide further insights into the scalability and performance improvements of our approach.

This work was generated by THE AI SCIENTIST (Lu et al., 2024).

### Figure 5: Missing plots and placeholders

In some cases, the placeholder filenames were shown in the PDF output instead of real images (see Figure 5). This made the results section appear incomplete or broken and reduced the credibility of the work. Only the baseline figure was correctly inserted, while all others remained empty frames with filenames.

## 4 CONCLUSION

We successfully ran five research pipelines with the AI Scientist and analyzed the resulting papers in detail. Each pipeline was based on either an original idea (e.g. adaptive early stopping, adaptive learning rate in SGD) or an AI-generated one (e.g. sparsity-inducing regularization). The experiments were executed on the MovieLens 100K dataset and each run led to a complete paper, including abstract, method, results and conclusion.

In terms of automation, the AI Scientist performed remarkably well. Once configured, it could autonomously generate ideas, run experiments and write full manuscripts with minimal user input. This level of hands-off research automation is impressive and confirms the system’s ability to reduce manual effort significantly. On average, one full pipeline took about 3 hours to complete and cost around €9 in API usage.

However, the generated outputs revealed clear limitations. All five papers showed consistent structural completeness but suffered from weak scientific depth. Most of the manuscripts included only a handful of citations, which are usually outdated and the ‘Related Work’ sections were generic or superficial. Explanations for algorithmic choices or hyper parameters were minimal or missing entirely. Moreover, none of the results included error bars, confidence intervals or repeated trials across different random seeds, limiting the reliability of the reported findings.

Code changes between iterations were minimal in the original study ( 8% per iteration, as reported by Beel et al.). In our experiments, we observed even fewer changes between steps. In many cases, the modified code was nearly identical to the previous version, often with only a line or variable name adjusted. This suggests that the system struggles with true adaptivity and iterative refinement. Instead of improving or exploring new directions, the AI often made superficial or cosmetic edits. Similarly, many generated ideas were simply slight variations of well-known techniques, presented as novel concepts.

These findings align with the original authors’ metaphor of the AI Scientist being similar to a “unmotivated undergraduate student rushing to meet a deadline”: producing papers that look plausible but lack strictness and depth. Still, the platform has clear strengths: it enables rapid prototyping, scales well with minimal effort and offers a glimpse into future research workflows where AI agents assist or accelerate early-stage idea exploration.

We did not explore multiple datasets or repeat experiments with different random seeds, mostly due to resource constraints. Additionally, we did not create our own algorithm templates, though doing so could have improved both the novelty and relevance of the outputs.

In summary, the AI Scientist performs surprisingly well as a research assistant but cannot yet replace the role of a human researcher. It lacks critical judgment, understanding of scientific context and methodological robustness. Future work should focus on improving these aspects, especially through better literature understanding, stronger evaluation mechanisms and clearer differentiation between real innovation and superficial novelty.

## 5 LIMITATIONS

There are several things we could have done to improve the quality and completeness of our evaluation. First, we could have created our own custom templates (e.g. KNN or SGD), but we decided against it due to time constraints. Second, we only used the MovieLens 100K dataset, even though testing the ideas on additional datasets would have made the results more robust. We also only generated a limited number of papers, generating more would have given us a better overall picture, but we were restricted by API costs. Finally, we did not test different random seeds or repeat the experiments multiple times, again due to cost limitations. These are all areas that could be explored in future work with more time and resources.

## 6 ACKNOWLEDGMENTS

This work was conducted as part of the Machine Learning Praktikum at the University of Siegen [2].

## REFERENCES

- [1] Joeran Beel, Min-Yen Kan, and Moritz Baumgart. 2025. Evaluating Sakana’s AI Scientist for Autonomous Research: Wishful Thinking or an Emerging Reality Towards ‘Artificial Research Intelligence’ (ARI)? *arXiv preprint arXiv:2502.14297* (2025). <https://arxiv.org/abs/2502.14297> arXiv v2, submitted 20 Feb 2025, revised 22 Feb 2025.
- [2] Joeran Beel and Lukas Wegmeth. 2025. Machine Learning Praktikum. *Universität Siegen* (2025).
- [3] Chris Lu, Cong Lu, Robert Tjarko Lange, Jakob Foerster, Jeff Clune, and David Ha. 2024. The AI Scientist: Towards Fully Automated Open-Ended Scientific Discovery. *arXiv preprint arXiv:2408.06292* (2024). <https://arxiv.org/abs/2408.06292> arXiv v3, submitted 12 Aug 2024, revised 1 Sep 2024.