Qualitative Analysis of Machine Learning Methods

Leon Behrens
Department of Physics
University of Basel
Klingelbergstrasse 82, 4056 Basel, Switzerland
leon.behrens@stud.unibas.ch

February 24, 2025

Abstract

Machine learning has become an essential tool in various fields, including physics, where it may be used to detect phase transitions from data, without prior knowledge of its structure. This study explores the Multi-Task Learning-by-Confusion (LBC) scheme to identify phase transitions. To evaluate the effectiveness of LBC in image-based analysis, three text-to-image generative models were used: Stable Diffusion 2.1, Stable Diffusion 3.5 medium, and FLUX.1 Schnell. Images were generated based on the prompt "Technology of the year θ" for years ranging from 1900 to 2050. A pretrained ResNet-50 was adapted for multi-class classification, using a custom loss function to mitigate class imbalance. The results demonstrate that LBC effectively identifies phase transitions in image distributions, with detecting three major transition points for Stable Diffusion 2.1, while identifying only two broader transitions for Stable Diffusion 3.5 and FLUX.1. This suggests that newer models produce more distinguishable images.

1 Introduction

Machine learning (ML) has revolutionized numerous fields, including computer vision and natural language processing, by enabling models to learn complex patterns from data [1, 3]. Traditional supervised learning approaches rely on well-annotated datasets and gradient-based optimization to minimize loss functions, thus guiding models toward accurate predictions [10, 11]. Recent advances have introduced alternative learning techniques that aim to improve generalization, robustness, and efficiency. An example is Learning-by-Confusion (LBC), which uses controlled perturbations to gain a better understanding and deeper learning [2, 4].

LBC varies from conventional gradient descent-based optimization by purposely misleading the model during training in return of improved robustness and generalization and deeper learning [5, 6]. Instead of solely minimizing a predefined loss function, LBC introduces misleading conditions that force the model to differentiate between genuine patterns and misleading cues [7, 8]. This technique has been shown to improve generalization performance, mitigate overfitting, and enhance robustness to distribution shifts [9]. LBC is especially useful in the detection of phase transitions form data, as it does not require prior knowledge of the underlying phases [5]. In this approach, a physical system is sampled at different values of a tuning parameter that determines the distribution of states. These states are then processed by an algorithm to identify the critical parameter values where the system undergoes a phase transition.

This paper focuses on the implementation of various image generation models and their analysis. During the analysis, phase transitions were detected using Multi-Task LBC. The image creation process was performed using Stable Diffusion 2.1 (SD21), Stable Diffusion 3.5 medium (SD35), and FLUX.1 Schnell (FX1), developed Stability AI and Black Forest Labs, respectively [13, 14, 16, 17, 19, 20]. SD21 was chosen as a baseline model to reproduce results from J. Arnold and N. Lörch [7]. The addition of SD35 and FX1 aimed to validate SD21's results and gain further understanding of the performance of different image creation models. Stable Diffusion 3.5 medium was selected over Stable Diffusion 3.5 and FLUX.1 Schnell over FLUX.1 Pro, due to their faster processing speed, reducing training time and conserving computational resources. [15, 18, 21]

For the analysis, a pretrained ResNet-50 was used and its last layer was modified using a custom loss function for unbiased, Multi-Task LBC. This loss function applies a weighted binary cross-entropy loss to correct for class imbalance [10]. These weights themselves were implemented with a custom class. Furthermore, a custom class was implemented to extract the labels for the training and evaluation from the generated images. Both the training loop and evaluation loop were customized to accommodate this approach.

The customized analysis combined with the hyperparameters listed in the Appendix A.1 trained the ResNet-50 over 151 epochs using the Adam optimizer [23]. Training times varied among models, with FX1 being the fastest, followed by SD21, and SD35 being the slowest, typically ranging from 9 to 13 hours depending on the model [22].

2 Theoretical Background

2.1 Multi-Task Learning in Machine Learning

Classical machine learning often employs Single-Task Learning (STL), where an artificial neural network (ANN) processes a single output task in isolation. Picture 1 shows STL with four separate ANNs, each trained independently on the same input data. Since these networks do not share parameters, knowledge learned by one model cannot transfer to another, limiting generalization across tasks.

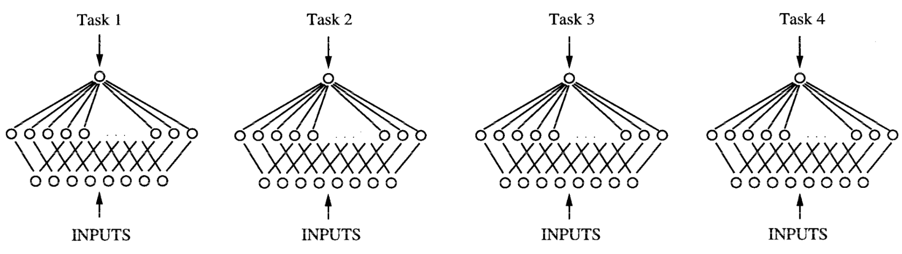

Figure 1: Single-Task Learning (STL) using Backpropagation of four tasks with the same inputs. [1]

Multi-Task Learning (MTL) addresses this limitation by enabling simultaneous learning of multiple related tasks. Instead of training independent ANNs, MTL enables generalization by utilizing the inter-task dependencies from the training of related tasks. This is done by training tasks in parallel while using a shared representation. Figure 2 illustrates a Multi-Task Backpropagation using the same inputs and tasks from Figure 1, but working with a single ANN. In effect, the training signals for the extra tasks serve as an inductive bias. By jointly optimizing related tasks, MTL acts as a form of implicit regularization, preventing overfitting and leading to more robust models.
MTL works by introducing a shared feature representation between tasks. This is achieved by (1) Hidden-Layer-Weight-Sharing, where the ANN learns the shared feature representations, forcing the ANN to extract generalizable patterns rather than task-specific ones, by (2) Auxiliary-Task-Learning, using the presence of additional tasks to enhance the learning of the primary tasks by providing supplementary training signals, and by (3) Regularization-Effect, where MTL reduces

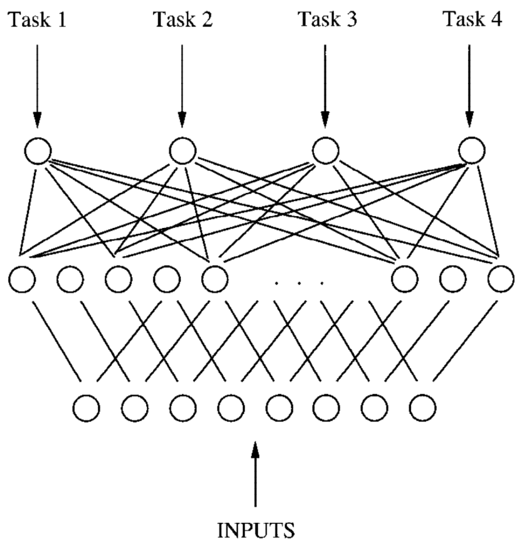

Figure 2: Multi-task Backpropagation (MTL) of four tasks (outputs) with the same inputs. [1]

overfitting by distributing the learning process among multiple tasks, resulting in a regularization technique. [1–4].

2.2 Learning-by-Confusion

Learning-by-Confusion (LBC) is a machine learning technique created to improve model generalization and robustness by adding intentional uncertainty into the training process. [5] Where as traditional training approaches exclusively focused on minimizing the loss function, LBC introduces intentional label shuffling or additional malicious conditions to challenge the model's ability to differentiate real patterns from misleading information.

The key idea behind LBC is that the model learns deeper structural patterns in the data when encountering high classification uncertainty. Thus, the ANN is forced to quit shortcuts. LBC operates by introducing controlled perturbations, ensuring that the model is exposed to mislabelled or misleading data before returning to correctly labelled training. Furthermore, the model's confidence levels are tracked as it learns to distinguish between real and misleading samples. Once the model reaches an optimal level of confusion, it is guided back to correctly labelled data, improving its robustness and preventing overfitting. [5–9]

2.3 Unbiased Learning-by-Confusion with Multi-Task Learning

Recent research has demonstrated that MTL can improve the effectiveness of LBC [7]. By training across multiple tasks simultaneously, the ANN is exposed to a wider range of data distributions, forcing it to learn meaningful representations rather than shortcuts. Key advantages of integrating MTL with LBC include improved generalization, faster convergence and better phase transition detection. Later means, that MTL enhances the ANN's ability to detect sharp transitions in data, improving classification accuracy.

In the simplest form of LBC, a (physical) system undergoes a phase transition as a function of a real value θ. To detect the critical point θ_C, where the phase transition occurs, the θ-axis is divided into $n+1$ different points. At each point N separate samples are taken. This results in n different ways to separate the θ-axis into two regions $\Theta_>$ and $\Theta_<$, where $\Theta^n_> = \{\theta|\theta > \theta_C\}$ and

$\Theta^n_< = \{\theta|\theta < \theta_C\}$. For each of these splitting a separate classifier is trained. I.e. the classifier n that achieves the lowest error rate on evaluation must have been trained closest around the natural splitting of the data. Thus, the classifier achieving the lowest error rate provides the best estimate $\tilde{\theta}_n$ for the critical transition point, as it most effectively distinguishes between pre- and post-transition states.

The loss function of the nth classifier is an unbiased binary cross-entropy loss given as:

$$\mathcal{L}_n = -\frac{1}{2} \sum_{y\in\{>,<\}} \frac{1}{|\mathcal{D}^y_n|} \sum_{\boldsymbol{x}\in\mathcal{D}^k_n} \log\left(\hat{p}(y|\boldsymbol{x})\right) \tag{1}$$

where $|\mathcal{D}^y_n|$ is the size of a given dataset $\mathcal{D}^y_n$ of samples $\boldsymbol{x}$ take from a region Θ_n and $\hat{p}$ is the estimated class probability of the classifier.

Analogously, the error rate of the nth classifier can be approximated as

$$p^{\text{err}}_n \approx \frac{1}{2} \sum_{y\in\{>,<\}} \frac{1}{|\mathcal{D}^y_n|} \sum_{\boldsymbol{x}\in\mathcal{D}^k_n} \text{err}_n(y, \boldsymbol{x}) \tag{2}$$

For each sample $\boldsymbol{x}$ the error $\text{err}_n(y, \boldsymbol{x}) \in [0, 1]$. The both extrema correspond to a correct classification or completely wrong classification, respectively. When training the ANN, $\mathcal{D}^y_n$ refers to the training dataset, where as in the evaluation (estimating the error) it refers to the evaluation dataset.

Combining this with MTL, not a new classifier is trained for every splitting n, but a single classifier with n outputs. The MTL loss function is $\mathcal{L} = \frac{1}{n}\sum_{i=0}^{n-1} \mathcal{L}_i$ and the error stays the same. [7]

2.4 Image Generation and Diffusion Models

Artificial image generation has been a primary topic in AI, especially with the recent begin of generative models such as Variational Autoencoders (VAEs) [10], Generative Adversarial Networks (GANs) [11], and more recently, Diffusion Models [12]. These models have enabled machines to generate highly realistic images.

2.4.1 StabilityAI: Stable Diffusion 2.1

Stable Diffusion 2.1 (SD21) was developed by StabilityAI in 2022 and is an advanced latent diffusion model designed for high-resolution image synthesis. As an improvement over earlier models, SD21 improves image clarity and coherence, particularly for detailed prompts.

Unlike earlier GAN-based approaches, SD21 uses Denoising Diffusion Probabilistic Models (DDPMs). The forward diffusion process gradually adds Gaussian noise to an image over T steps, degrading it step by step:

$$q(x_t|x_{t-1}) = \mathcal{N}\left(x_z;\ \sqrt{1-\beta_t}\cdot x_{t-1},\ \beta_t \cdot I\right) \tag{3}$$

where $q(x_t|x_{t-1})$ represents the forward diffusion process and is modelled as a Gaussian distribution, x_t is the image at time t, representing the progressively noisier version of the image and β_t controls the noise variance.

In the reverse process, a deep neural network (DNN), learns how to remove the noise step by step. In this denoising process, a U-Net is typically trained to estimate and remove the added noise. The probabilistic formulation of this process is given by:

$$p_\theta(x_{t-1}|x_t) = \mathcal{N}\left(x_{t-1};\ \mu_\theta(x_t, t),\ \Sigma_\theta(x_t, t)\right) \tag{4}$$

μ_θ is the predicted mean of the distribution, which guides the denoising process and Σ_θ is the variance used for noise estimation. The DNN learns both of these functions and predicts, how much noise must be removed in each step.

SD21 first compresses images into a lower-dimensional latent space representation using a VAE. This mapping of pixel-space images into latent space representation before applying diffusion, reduces computational complexity allowing for for more efficient diffusion-based denoising. After the diffusion process is completed, a VAE decoder reconstructs the final image from the latent space. To ensure accurate text-to-image alignment, SD21 conditions the denoising process on CLIP embeddings, which guide the model to generate images that closely match the input prompt [13–15].

2.4.2 StabilityAI: Stable Diffusion 3.5 Medium

Stable Diffusion 3.5 Medium (SD35) is an iterative improvement over SD21 created in 2024. It builds on DDPMs, but introduces score-based generative modeling, where it uses a learned score function $s_\theta(x_t, t)$ to approximate the gradient of the data distribution in eq. 5.

$$\nabla_x \log\left(p_t(x)\right) = \frac{p(x_t|x_0) - x_t}{\sigma_t^2} \tag{5}$$

where σ_t is the true variance at time t.

Additionally, SD35 introduces an adaptive noise schedule for smoother denoising steps and classifier-free guidance, improving text-to-image consistency compared to SD21. The noise levels β_t are dynamically adjusted (eq. 6) ensuring smoother denoising. The classifier-free guidance utilises a score function, which is updated using a mixing coefficient ω, allowing the model to balance text guidance and diversity, thus improving image flexibility (eq. 7).

$$\beta_t = \beta_{\min} + (\beta_{\max} - \beta_{\min}) \cdot \left(\frac{t}{T}\right)^s \tag{6}$$

$$\tilde{s}_\theta(x_t, t) = \omega \cdot s_\theta^{\text{cond}}(x_t, t) + (1 - \omega) \cdot s_\theta^{\text{uncond}}(x_t, t) \tag{7}$$

SD35 still applies diffusion in pixel space, using a VAE for latent encoding and decoding. However, the refinements in noise scheduling and classifier-free guidance improve both image fidelity and prompt adherence [16–18].

2.4.3 Black-Forest-Labs: FLUX.1 Schnell

FLUX.1 Schnell (FX1) is an high-speed diffusion model created by Black-Forest-Labs in 2024. It is optimized for ultra-fast image synthesis, making it significantly faster than Stable Diffusion models. Unlike SD21 and SD35, which perform diffusion in pixel space, FX1 operates entirely in latent space. This allows for lower computational costs while maintaining image quality through adaptive noise scaling and physics-guided regularization.

The image transformation is modelled as:

$$z_t = f_\phi(x_t), \tag{8}$$

where $f_\phi(x_t)$ describes a deep encoder function that maps a high-dimensional image space to a lower-dimensional latent representation. Additionally, FX1 uses dynamically adjusted noise levels with a learned scaling factor α_t and adaptation factor σ_t:

$$p(x_t|x_{t-1}) = \mathcal{N}(x_t;\ \alpha_t x_{t-1},\ \sigma_t I) \tag{9}$$

Instead of blindly denoising images, the model considers real-world physical properties relevant to the generated data, in the diffusion process.

$$q(x_t|x_{t-1}) = \mathcal{N}(x_t;\ \mathcal{F}(x_t, \Theta),\ \beta_t I) \tag{10}$$

$\mathcal{F}$ contains the physics-related constraints, such as such as fluid dynamics, material deformation, or lighting consistence, and Θ represents tunable hyperparameters for optimizations. Furthermore, FX1 decreases the denoising steps by introducing a dynamically adjustable constant λ_t that updates the step-size for faster convergence.

$$x_{t-1} = x_t - \lambda_t s_\theta(x_t, t) \tag{11}$$

However, FX1's focus on speed results in minor texture loss compared to full-resolution models like Stable Diffusion. [19–21]

3 Application

The three analysed text-to-images modes were StabilityAI's Stable Diffusion 2.1 (SD21) and Stable Diffusion 3.5 medium (SD35) and Black-Forest-Labs' FLUX.1 Schnell (FX1) [22]. Using the hyperparameters from Table 2, images are sampled for each integer $\theta \in [1900, 2050]$ with the prompt "Technology of the year θ".

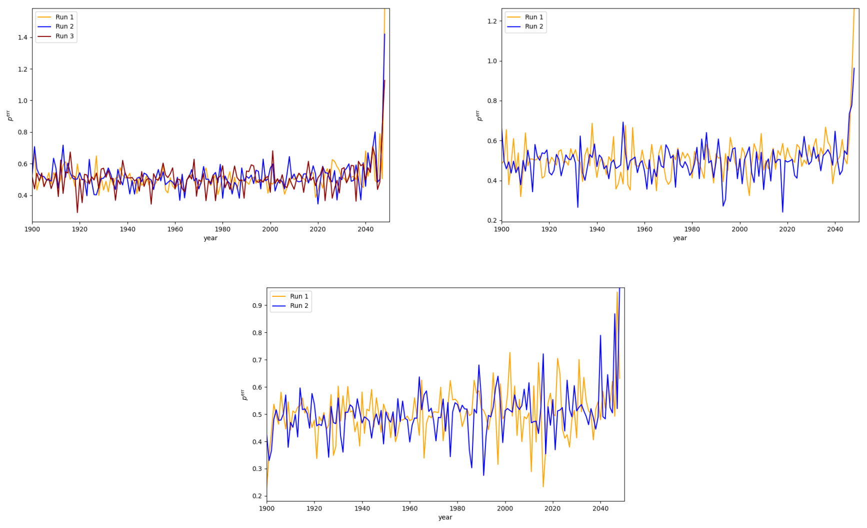

Figure 3: (a) Analysis of the SD21 images after one training epoch for three different runs. In each run, the weights of the ResNet-50's last layer were initialized differently, evident as no lines are identical. (b) shows the analysis of the SD35 images after epoch 1 and (c) the analysis of the FX1 images.

While the generated datasets do not feature a phase transition in the physical sense, the learning-by-confusion scheme can still be used to identify points in the parameter space where the data distribution changes rapidly [7]. To evaluate the generated datasets, I used a pretrained ResNet-50 model [25] model from PyTorch [26] and modified its last layer for classification. Since the dataset contained images spanning 151 different years, the network was adapted to have 151 output nodes. The implementation details are provided in Appendix A.1. The number of images generated per model and per year, along with the number of analysis runs, is summarized in Table 1:
Figure 3 shows that the weights of ResNet-50's last layer were initialized differently for each run across all models. This is evident as none of the graphs display identical lines, confirming that each run had an independent initialization.

Model	Nr. of Images per Year	Nr. of Analysis Runs
SD21	81	3
SD35	30	2
FX1	30	2

Table 1: SD21 was analysed in three separate runs, while SD35 and FX1 were each analysed in two runs. The used hyperparameters are listed in the Appendix A.1 in the Table 2.

Figure 4 illustrates the validation loss for all analysis runs. A strong decrease in validation loss is observed in the early epochs, followed by convergence to a minimal intrinsic error after epoch 40. Individual validation loss curves for each model are shown in Figures 7, 9 and 11 (Appendix A.2).

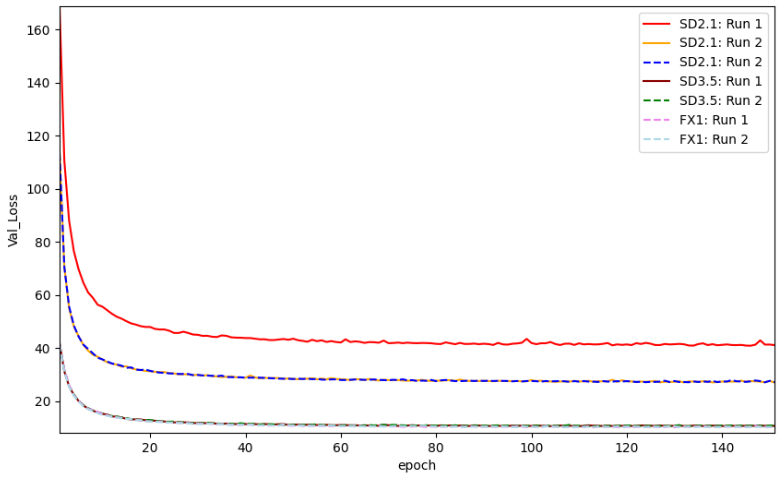

Figure 4: Validation loss of the seven different runs using all models with hyperparameters from Table 2.

Figure 5 shows the validation error of the 151st (and last) epoch. Each model is averaged over its runs (ref. table 1) and shown with a solid line. The standard deviation is added as the area around the line. The orange line represents the ST21, the blue line SD35 and the red line represents FX1.

For ST21, the graph shows at least three major local minima, indicating rapid changes in the image dataset. The first occurs around 1929, the second is broader and appears in the late 1990s, and the third occurs between 2020 and 2022. However, SD35 and FX1 show only two major local minima, one from 1930 to 1979 and a second, smaller one from 1999 to 2014. Separate graphs for each model are in the Appendix A.2 in Figures 8, 10, 12.

Lastly, the standard deviation of the RBG values of each pixel in an image averaged over all images of the same year is shown in Figure 6 for SD21. The graph does not show a clear trend but on average three different plateaus are observable, form 1900 to the 1920s, from the 1920s to the 1960s and from the 1970s to the late 1990s. Afterwards, the graph strongly decreases.

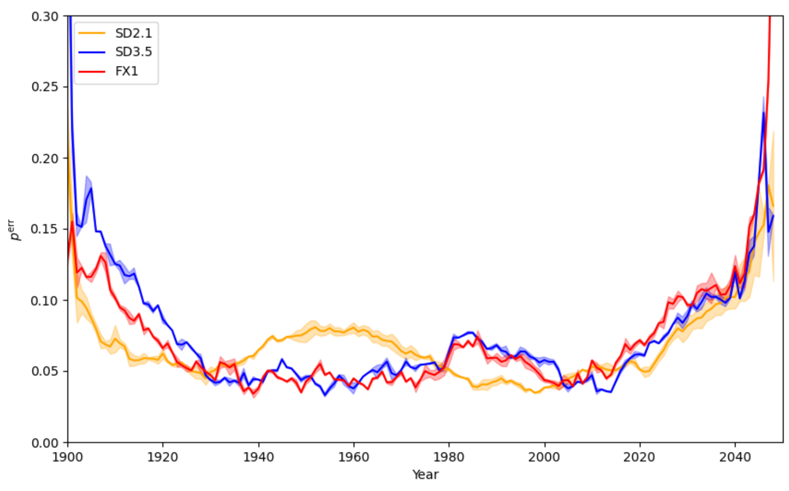

Figure 5: Error rate by using multi-task learning-by-confusion for the different datasets as a function of θ in the prompt "Technology of the year θ". The coloured lines show the results from three different models, averaged over their runs with different hyperparameters (Table 2) as described in the Appendix A.1. The solid lines represents the mean and the area the standard deviation.

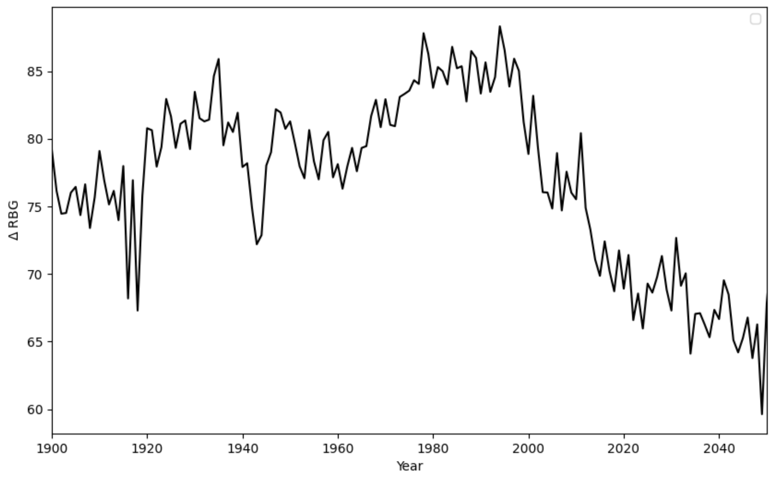

Figure 6: Shows the standard deviation of the RBG values of each pixel in an image, generated by DS21, and averages over each year. A value close to zero indicates a grey images or a single colour images. The graph does not show a clear trend but on average three different plateaus are observable, form 1900 to the 1920s, from the 1920s to the 1960s and from the 1970s to the late 1990s.

4 Discussion and Conclusion

This study shows the use of Learning-by-Confusion (LBC) within Multi-Task learning (MTL) in analysing image datasets and allowed for the identification of phase transitions within the dataset. It demonstrates how LBC can be utilized to uncover phase transitions in the generated images datasets, especially when no data about the underlying phase is given. The successful training of

Multi-Task LBC is confirmed in Figure 4, where each run exhibits the typical pattern of a strong decrease in validation loss during the early training stages, followed by convergence to a minimal value at higher epoch numbers (after epoch 40).

From Figure 4, it follows that both newer models (SD35 and FX1) generate higher-quality, more structurally consistent, and more distinguishable images compared to SD21. This is evident from the Validation Loss, where SD21's values seem to converge at approximately 50 and 35 for analysis run 1 and runs 2-3, respectively, while SD35 and FX1 converge around 12 (see Appendix A.2 Fig. 7, 9 & 11). The difference in Validation Loss values for SD21 is due to the variation in the training-to-validation dataset ratio. Run 1 used an 80:20 ratio, while Runs 2 and 3 used a 70:30 ratio. Additionally, it follow from Figure 4, that a small change in the learning rate does not significantly affect Validation Loss, as Run 2 and Run 3 exhibit almost identical convergence. Run 2 used a learning rate of $0.8 \cdot 10^{-4}$ and Run 1 and Run 3 used a learning rate of $0.5 \cdot 10^{-4}$.

A key observation were the three major phase transition of SD21, happening from 1929-1930, in the 1990's and from 2020 to 2022, reproducing the results from J. Arnold and N. Lörch [7]. The reason for the first minima might be the change from gray images to colour images. However, Figure 6, showing the average standard deviation of the RBG values per year, does not support that statement, even though there is a plateau from 1900 to the 1920s and a higher plateau from the 1920s to the 1960s. The decrease in Figure 6 after 2022 might be linked to advancements in mobile technology and the rise of futuristic imagery, both of which often feature a higher proportion of blue colours.

The second minima (1990s) could be associated with the invention of the internet and its influence on digital imagery, though no specific analysis was conducted to support this claim. The final dip around 2020 may result from SD21's training data, as it was primarily trained on pre-2022 images from LAION-5B [24].

SD35 and FX1 captured only two local minima, a broad one from 1930 to 1979 and a second, smaller one from 1999 to 2014. This suggests that newer models generate more distinguishable images, making them more versatile and reducing abrupt transitions, which indicates an improvement in newer model architectures. Moreover, the analysis was performed using ResNet-50, a large, pre-trained model capable of effectively mapping images to their corresponding years. It is possible that a less powerful model might reveal phase transitions closer to those seen in SD21, or intensity the current peaks and dips. Additionally, no significant differences between SD35 and FX1 were observed. This might be because full versions of these models were not used in this study, limiting the ability to distinguish their subtle variations.

In conclusion, the implementation of different image generation models and their analysis using Multi-Task Learning-by-Confusion was a success, as phase transitions in the different image datasets are observable.

For future work, it is recommended to use the full versions of SD35 and FX1, generate larger image datasets, and explore alternative analysis models to better distinguish differences between these two models. This may be achieved by varying the classification model for the phase parameter θ, or by introducing an alternative classification approach, where the content of the images is analysed, providing deeper insights into the underlying reasons for the observed phase transitions.

Acknowledgments

I would like to thank my tutors, Julian Arnold and Niels Lörch, for their invaluable support, guidance and expertise throughout this work and the provision of their Python Code. Additionally, I extend my gratitude to Christoph Bruder and the Bruder Group for the possibility to work on the topic and their financial support, which enabled my computations on the SciCore Cluster.

References

[1] Caruana, R. (1997). Multitask Learning, Machine Learning, 28

[2] Baxter, J. (2000). A Model of Inductive Bias Learning. arXiv:1106.0245

[3] Ruder, S. (2017). An Overview of Multi-Task Learning in Deep Neural Networks. arXiv:1706.05098

[4] Zhang, Y., & Yang, Q. (2021). A Survey on Multi-Task Learning. arXiv:1707.08114

[5] van Nieuwenburg, E. P. L., Liu, Y.-H., & Huber, S. D. (2017). Learning Phase Transitions by Confusion. Nature Physics, 13, 435–439

[6] Richter-Laskowska, M., Kurpas, M., & Maśka, M. (2023). Learning by Confusion Approach to Identification of Discontinuous Phase Transitions. Physical Review E, 108, 024113

[7] Arnold, J. & Lörch, N. (2023). Fast Detection of Phase Transitions with Multi-Task Learning-by-Confusion. arXiv:2311.09128

[8] Song Sub Lee (2019). Confusion Scheme in Machine Learning Detects Double Phase Transitions and Quasi-Long-Range Order. Phys. Rev. E 99, 043308

[9] Berthelot, D., et all. (2019). MixMatch: A Holistic Approach to Semi-Supervised Learning. arXiv:1905.02249

[10] Kingma, D. P., & Welling, M. (2013). Auto-Encoding Variational Bayes. arXiv:1312.6114

[11] Goodfellow, I., Pouget-Abadie, J., Mirza, M., et al. (2014). Generative Adversarial Networks. arXiv:1406.2661

[12] Ho, J., Jain, A., & Abbeel, P. (2020). Denoising Diffusion Probabilistic Models. arXiv:2006.11239

[13] Stability AI (2022). Stable Diffusion 2.1 Release Announcement.

[14] Stability AI (2022). Stable Diffusion 2.1 Model Repository. Hugging Face.

[15] Ramesh, A., Dhariwal, P., Nichol, A., et al. (2022). Hierarchical Text-Conditional Image Generation with CLIP Latents. arXiv:2204.06125

[16] Stability AI (2024). Introducing Stable Diffusion 3.5, Official Announcement.

[17] Stability AI (2024). Stable Diffusion 3.5 Medium Model Repository. Hugging Face.

[18] Song, Y., Meng, C., & Ermon, S. (2021). Denoising Diffusion Implicit Models. ICLR. arXiv:2010.02502

[19] Black Forest Labs (2024). FLUX.1 Schnell – Official Page.

[20] Black Forest Labs (2024). FLUX.1 Schnell Model Repository. Hugging Face.

[21] Luo, T., Zhao, H., & Wang, Z. (2023). High-Fidelity Physics Simulations Using Diffusion Models. arXiv:2301.01234

[22] Behrens, L. (2024). "Python implementation of multi-task learning-by-confusion and image creation using Stable Diffusion 2.1, Stable Diffusion 3.5 medium and FLUX.1 schnell". GitHub Repository

[23] Kingma D. P. & Ba, J. (2014). Adam: A Method for Stochastic Optimization. arXiv:1412.6980

[24] Schuhmann, C. & Beaumont, R. et al. (2022). LAION-5B: An open large-scale dataset for training next generation image-text models. Adv. Neural Inf. Process. Syst. 35, 25278.

[25] He, K., Zhang, X., Ren, S. & Sun, J. (2016). Deep Residual Learning for Image Recognition. Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.

[26] Paszke, A. et al. (2019) PyTorch: An Imperative Style, High-Performance Deep Learning Library. NeurIPS Proceedings

A Appendix

A.1 Implementation

The Python implementation of the image creation using Stable Diffusion 2.1, Stable Diffusion 3.5 medium and FLUX.1 Schnell, as well as the implemention of the multi-task leaning-by-confusion analysis is available at [22]. The hyperparameters used to generate the figures in this article are summarized in Table 2. Here, N_{train} and N_{eval} refer to the number of images per year within the training and validation set, respectively. For training, we use the Adam optimizer [23] with various hyperparameters listed in Table 2.

Model	Run Nr.	N_{train}	N_{eval}	Ratio	Learning Rate	Batch Size	Training Epochs
ST21	1	9'720	2'430	80:20	$0.5 \cdot 10^{-4}$	1'024	151
ST21	2	8'505	3'645	70:30	$0.8 \cdot 10^{-4}$	1'024	151
ST21	3	8'505	3'645	70:30	$0.5 \cdot 10^{-4}$	1'024	151
ST35	1	3'600	900	80:20	$0.5 \cdot 10^{-4}$	1'024	151
ST35	2	3'600	900	80:20	$0.5 \cdot 10^{-4}$	1'024	151
FLUX.1	1	3'600	900	80:20	$0.5 \cdot 10^{-4}$	1'024	151
FLUX.1	2	3'600	900	80:20	$0.5 \cdot 10^{-4}$	1'024	151

Table 2: Hyperparameters used in this paper. SD21 refers to Stable Diffusion, Version 2,1; SD35 refers to Stable Diffusion, Version 3.5 medium; FLUX.1 refers to FLUX.1 Schnell. A hyperparameter not adjusted manually, but different for every run, is the random initialization of the weights in the ResNet-50's last layer, as shown in Figure 3

A.2 Application

A.2.1 Stable Diffusion 2.1

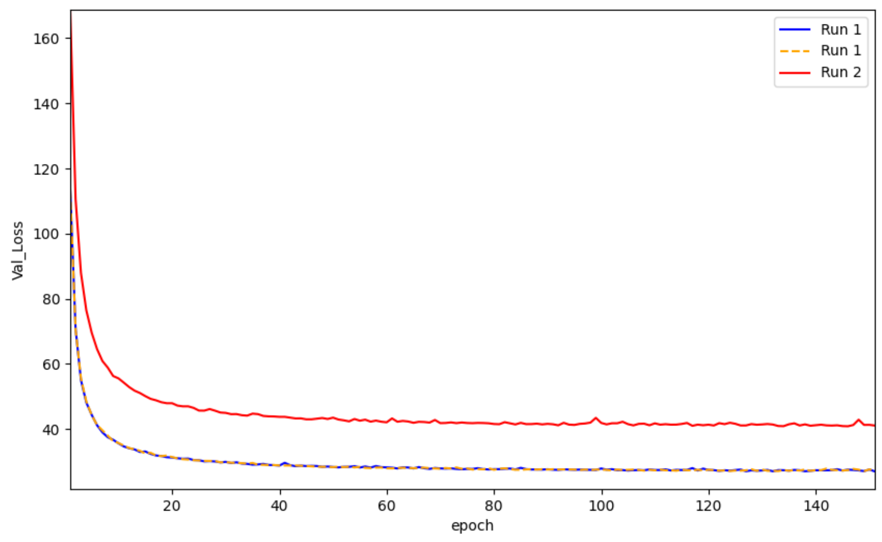

Figure 7: Validation loss per training epoch of the three different runs with the SD21 model and the hyperparameters in Table 2, where 81 images per year were generated.

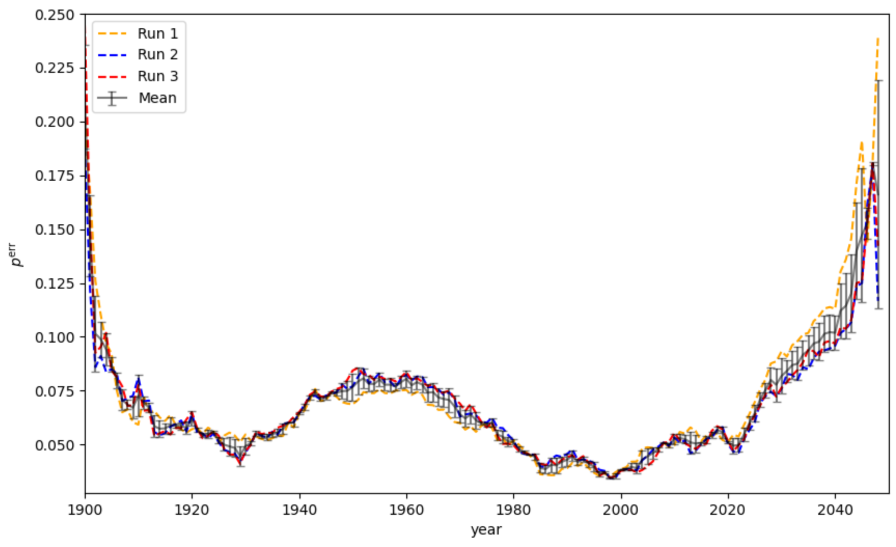

Figure 8: Error rate by using multi-task learning-by-confusion for the SD21 dataset as a function of θ in the prompt "Technology of the year θ". The coloured lines show the results from three different runs with different hyperparameters (Table 2) as described in the Appendix A.1. The gray line represents their mean and standard deviation. The graph shows three local minima, the first on from 1929 to 1930, the second one in the 1990s and the last on from 2020 to 2022.

A.2.2 Stable Diffusion 3.5 medium

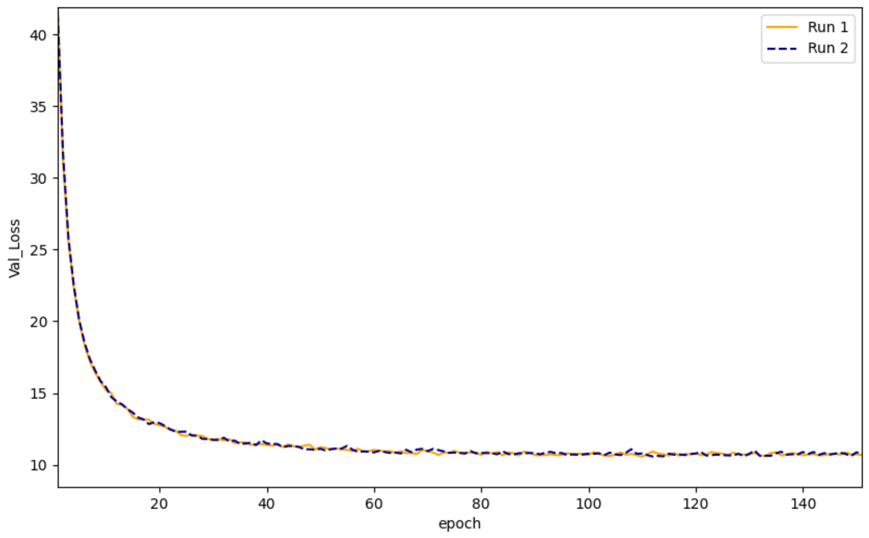

Figure 9: Validation loss per training epoch of the three different runs with the SD35 model and the hyperparameters in Table 2, where 30 images per year were generated.

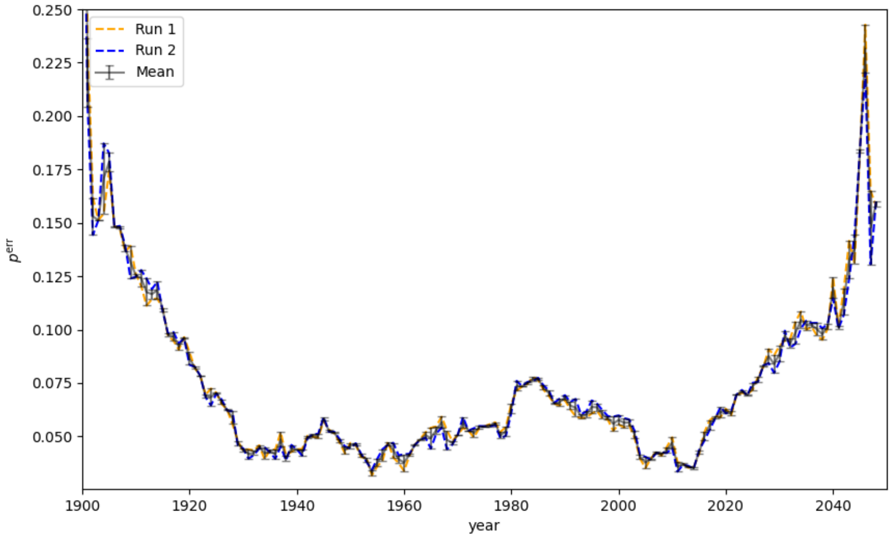

Figure 10: Error rate by using multi-task learning-by-confusion for the SD35 dataset as a function of θ in the prompt ”Technology of the year θ”. The coloured lines show the results from three different runs with different hyperparameters (Table 2) as described in the Appendix A.1. The gray line represents their mean and standard deviation. The graph show two local minimal a broad on from 1930 to 1979, and a second on from 1999 to 2014.

A.2.3 FLUX.1 Schnell

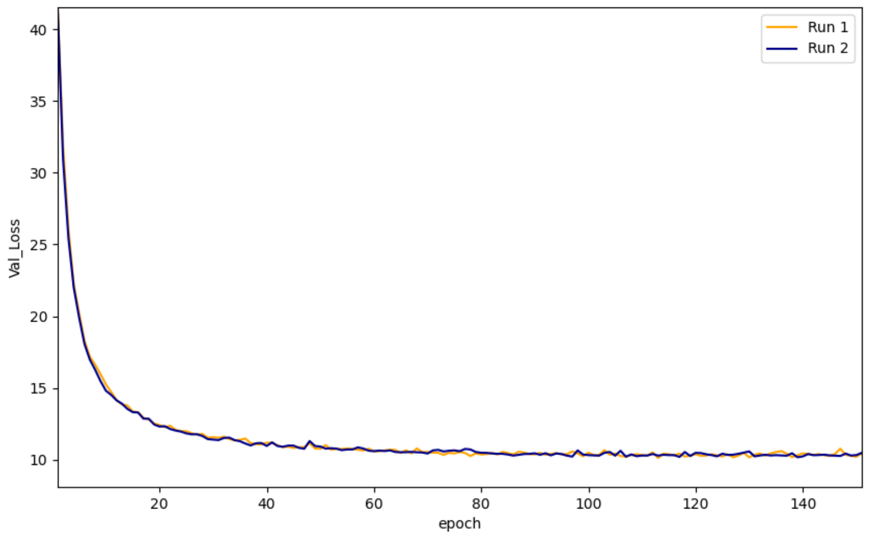

Figure 11: Validation loss per training epoch of the three different runs with the FX1 model and the hyperparameters in Table 2, where 30 images per year were generated.

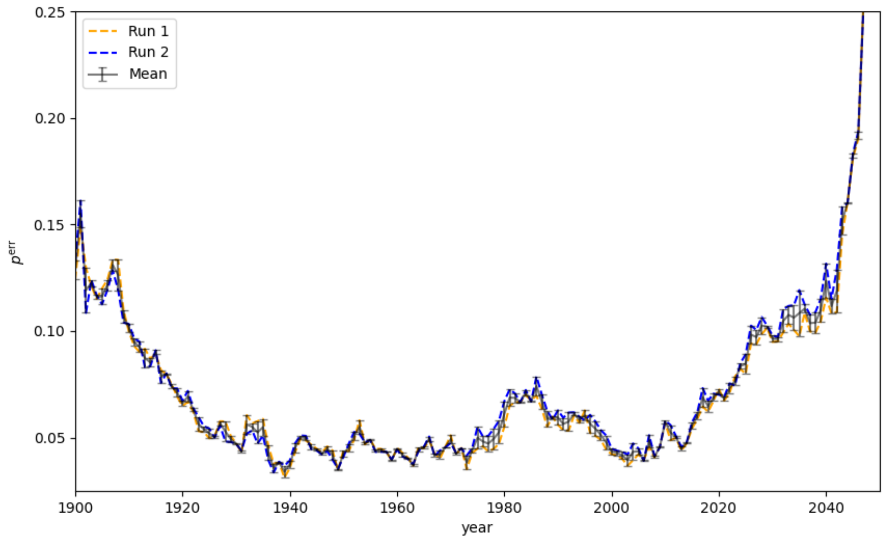

Figure 12: Error rate by using multi-task learning-by-confusion for the FX1 dataset as a function of θ in the prompt "Technology of the year θ". The coloured lines show the results from three different runs with different hyperparameters (Table 2) as described in the Appendix A.1. The gray line represents their mean and standard deviation. The graph show two local minimal a broad on from 1930 to 1979, and a second on from 1999 to 2014.