Machine Learning
Assignment 11

CHANG Yung-Hsuan (張永璿)

111652004

eiken.sc11@nycu.edu.tw

November 21, 2025

1. Summarize all your "unanswered questions" in previous assignments. Ask AI whether there are corresponding researches for each question. If so, list reference; if not, describe the problem clearer so that it is more understandable.

   a. (*Original Question.*) I heard something related to "momentum" a few years ago during some conversation about the gradient descent method. If I remember correctly, it should be a tool that accelerate the convergence of the gradient descent method. However, it seems to lose popularity these days. Is it my misconception or just the method is not suitable for us to learn?

      Answer. This is NOT an open problem.

      See [1] for the origin of momentum and [2] for the Adam optimizer.

   b. (*Original Question.*) How efficient does a hidden layer perform to approximate a monomial? Fix the number of data points to a constant or a scalar times the power of the monomial. Do we have a performance benchmark as we evaluate sorting algorithms to evaluate the effectiveness of a one-hidden layer neural network?

      Answer. This is NOT an open problem.

      See [3] for the efficiency.

   c. (*Original Question.*) Will we investigate CNN in this class? For example, how does convolution work effectively?

Answer. This is NOT an open problem.

See [4] for the details about unrolling convolution into a matrix-matrix multiplication to achieve high speeds on GPUs and [5] for Winograd's minimal filtering algorithm. The latter is a key algorithm inside NVIDIA's cuDNN library today.

d. (*Original Question.*) The MNIST dataset is mentioned during the lecture. It is known that there are CNNs to solve this kind of problem of recognizing images (or classifying images). My idea is naive, but I wonder if we consider those images points in a space (e.g., in $\mathbb{R}^{28\times 28}$) and cluster them, does this clustering do the same with classification?

Answer. This is NOT an open problem. To be more specific, the problem is related to manifold learning.

See [6] for reducing dimensions by using an autoencoder and [7] for using clustering instead of classification.

e. (*Original Question.*) How do we calculate the maximum likelihood estimators for those $\Sigma_1 \neq \Sigma_2$? Do we really calculate them or we just code so that we see the results in practice?

Answer. This is NOT an open problem.

See [8] for the derivation.

f. (*Original Question.*) I am curious about the role of regularization in neural networks. In linear models, ridge regression is commonly used to mitigate overfitting by penalizing large coefficients. I wonder whether similar techniques exist for neural networks—methods that modify the loss function to constrain or regulate the magnitude of weights within a reasonable range. Or, are such models primarily guided by empirical tuning of architectures and parameters rather than explicit analytical regularization?

Answer. This is NOT an open problem.

See [9] for the "Ridge" concept directly to neural networks and [10] for the dropout technique to break the co-adaptation of feature detectors and serves as a powerful form of regularization.

g. (*Original Question.*) In the denoising score matching, the data are assumed to follow

$$y_i = f_\theta(x_i) + \varepsilon_i, \quad \varepsilon_i \sim \mathcal{N}\left(0, \sigma^2 I\right).$$

Typically, we fix σ when fitting the model, but in practice, σ controls how much noise we assume in the data. A small σ means we trust the data more, leading to possible overfitting, while a large σ means we treat the data as noisier, often resulting in smoother and more generalizable models.

This made me wonder: instead of fixing a single σ, could we take a range of σ values and average the resulting models? Intuitively, this would mean combining models trained under different assumptions about data noise, which resembles the idea of *Bayesian model averaging*—integrating over different possible noise levels rather than committing to one.

Mathematically, this could be viewed as

$$\hat{f}(x) = \frac{1}{N_\sigma} \sum_{j=1}^{N_\sigma} f_{\theta(\sigma_j)}(x),$$

where each $f_{\theta(\sigma_j)}$ is a model trained under a different assumed noise variance σ_j^2.

Conceptually, this approach might yield a more robust estimate of $f_\theta(x)$, especially when the true noise level is uncertain or varies across the input space. I find this question meaningful because it connects the practical side of the denoising score matching, where we can directly control the level of noise, with theoretical ideas from probabilistic modeling and Bayesian inference.

Answer. This is NOT an open problem. This is the exact premise of the noise conditional score networks (NCSN).

See [11] and [12] written by Song about NCSN and continuous noise.

h. (*Original Question.*) In class, it was mentioned that sampling x_0 from p would satisfy the Fokker–Planck equation. I hope to gain a clearer understanding next week when we see the proof. For now, I wonder what kinds of probability densities actually satisfy a given Fokker–Planck equation.

To be more precise, for which drift f and diffusion G will the solution $p(x, t)$ of

$$\frac{\mathrm{d}p}{\mathrm{d}t} = \frac{\sigma^2}{2} \cdot p_{xx}$$

converge to a Gaussian distribution as $t \to \infty$?

In some special cases, such as pure diffusion or the Ornstein–Uhlenbeck process, the solution indeed converges to a normal distribution, as mentioned in class. I am curious whether we can characterize all families of solutions that remain invariant or converge to a stationary Gaussian distribution.

Answer. This is NOT an open problem and is a solved problem in statistical mechanics.

See [13] for the proof that the stationary distribution is determined by the potential of the drift force.

i. (*Original Question.*) We mentioned morphing in the lecture. I wonder how morphing with structures works; for example, mapping nose to nose, eyes to eyes, and mouth to mouth. In this case, does the probability-flow ordinary differential equation still apply?

Answer. This is NOT an open problem.

See [14] and [12] for the deterministic non-Markovian sampling process and for how the probability-flow ordinary differential equation is related to the curve that minimizes the kinetic energy of the data flow, effectively acting as a form of optimal transport that aligns data structures naturally.

2. Build a solvable model in your final project.

Modeling. We model the airport gate assignment problem as a 1D Bin Packing Problem, where Bins (Gates) are resources with fixed capacity $C = 1$ (time availability), and items (flights) are tasks with specific durations that need to be packed into bins.

The core challenge is the gap between planning and reality. A deterministic algorithm like Best-Fit Decreasing (BFD) packs flights tightly to minimize gate usage. However, in a stochastic environment where flight durations fluctuate, a tight schedule has zero margin for error. A slight delay causes an overflow (gate conflict), rendering the plan infeasible.

Thus, we propose a Stochastic Bin Packing Model where the agent does not just pack items, but actively manages risk by determining optimal safety margins.

For each flight i, the system observes a tuple (s_i, σ_i), where s_i is the nominal duration (scheduled time), and σ_i is the volatility score. The actual duration of the flight S_i is unknown during planning but follows a distribution parameterized by

$$\ln S_i \sim \mathcal{N}(\ln s_i, {\sigma_i}^2).$$

The model will output a specific safety padding $\beta_i = f_\theta(s_i, \sigma_i)$ for each flight, and the effective size used for planning becomes $\hat{s}_i = s_i + \beta_i$.

This formulation allows us to use standard heuristics within a learning loop, making the problem solvable without requiring complex real-world datasets.

We utilize the Best-Fit Decreasing (BFD) algorithm to pack the items based on their effective sizes $\hat{s}_i$. This leverages the efficiency of BFD while injecting AI-driven risk management.

We simulate the actual outcome by sampling S_i from the noise distribution defined by σ_i. Then, we calculate the cost based on the realization. We seek to find parameters θ to minimize the expectation of

$$\#\text{bins}(\hat{\mathbf{s}}) + \lambda \cdot \sum_j \max\left(0, \sum_{i \in \text{bin}_j} S_i - C\right), \tag{1}$$

where the first term forces the model to be efficient (don't add too much padding), and the second term penalizes the total overflow (don't add too little padding).

This model is the perfect "simplification" for a first step because it requires no external data, is mathematically sound, and is trainable.

In short, we have the following summary for this solvable problem.

Table 1. Ingredients.

Data	Synthetic
Tools	Combinatorial Heuristics
Environment	Stochastic Simulation

Table 2. Reinforcement Learning Paradigm.

Data Source	$\{(s_i, \sigma_i)\}$
Target Signal	The loss function (1)
Feedback Loop	Delayed feedback

Bibliography

[1] B. T. Polyak, “Some methods of speeding up the convergence of iteration methods,” *USSR Computational Mathematics and Mathematical Physics*, vol. 4, no. 5, pp. 1–17, 1964, [Online]. Available: https://www.sciencedirect.com/science/article/abs/pii/0041555364901375

[2] D. P. Kingma and J. Ba, “Adam: A Method for Stochastic Optimization,” in *3rd International Conference on Learning Representations (ICLR)*, 2015. [Online]. Available: https://arxiv.org/abs/1412.6980

[3] S. Liang and R. Srikant, “Why Deep Neural Networks for Function Approximation?,” in *International Conference on Learning Representations (ICLR)*, 2017. [Online]. Available: https://arxiv.org/abs/1610.04161

[4] K. Chellapilla, S. Puri, and P. Simard, “High performance convolutional neural networks for document processing,” in *Tenth International Workshop on Frontiers in Handwriting Recognition*, 2006. [Online]. Available: https://inria.hal.science/inria-00112631/document

[5] A. Lavin and S. Gray, “Fast algorithms for convolutional neural networks,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 4013–4021. [Online]. Available: https://arxiv.org/abs/1509.09308

[6] G. E. Hinton and R. R. Salakhutdinov, “Reducing the dimensionality of data with neural networks,” *Science*, vol. 313, no. 5786, pp. 504–507, 2006, [Online]. Available: https://www.science.org/doi/10.1126/science.1127647

[7] J. Xie, R. Girshick, and A. Farhadi, “Unsupervised deep embedding for clustering analysis,” in *International Conference on Machine Learning*, 2016, pp. 478–487. [Online]. Available: https://arxiv.org/abs/1511.06335

[8] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, 2nd ed. Springer, 2009. [Online]. Available: https://hastie.su.domains/ElemStatLearn/printings/ESLII_print12.pdf

[9] A. Krogh and J. A. Hertz, “A simple weight decay can improve generalization,” in *Advances in Neural Information Processing Systems*, Morgan-Kaufmann, 1992, pp. 950–957. [Online]. Available: https://proceedings.neurips.cc/paper/1991/file/8eefcfdf5990e441f0fb6f3fad709e21-Paper.pdf

[10] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Dropout: A simple way to prevent neural networks from overfitting,” *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014, [Online]. Available: https://www.jmlr.org/papers/volume15/srivastava14a/srivastava14a.pdf

[11] Y. Song and S. Ermon, “Generative modeling by estimating gradients of the data distribution,” in *Advances in Neural Information Processing Systems*, 2019. [Online]. Available: https://arxiv.org/abs/1907.05600

[12] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole, “Score-Based Generative Modeling through Stochastic Differential Equations,” in *International Conference on Learning Representations*, 2021. [Online]. Available: https://arxiv.org/abs/2011.13456

[13] H. Risken, *The Fokker-Planck Equation: Methods of Solution and Applications*, 2nd ed. in Springer Series in Synergetics. Springer, 1996. [Online]. Available: https://link.springer.com/book/10.1007/978-3-642-61544-3

[14] J. Song, C. Meng, and S. Ermon, “Denoising Diffusion Implicit Models,” in *International Conference on Learning Representations*, 2021. [Online]. Available: https://arxiv.org/abs/2010.02502