MIDDLE EAST TECHNICAL UNIVERSITY

MMI714
Generative Models for Multimedia

Introduction

Welcome

This is MMI714
“Generative Models for Multimedia”

This advanced deep learning course offers a comprehensive introduction to the principles and practice of generative modeling.

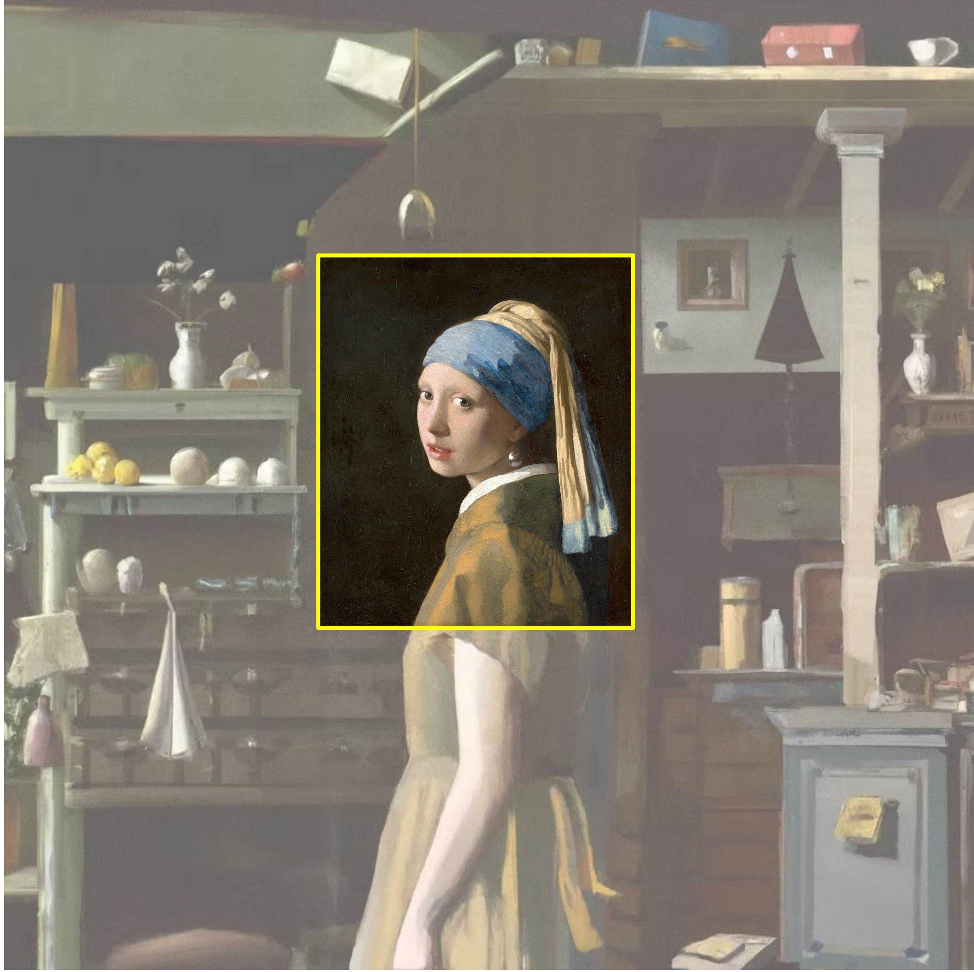


Image by openai.org

Welcome

This is MMI714
"Generative Models for Multimedia"

Beginning with a review of the mathematical foundations required for the course, students will gain an understanding of the conventional autoregressive methods used in generative modeling, as well as more contemporary techniques such as deep generative neural models and diffusion models.

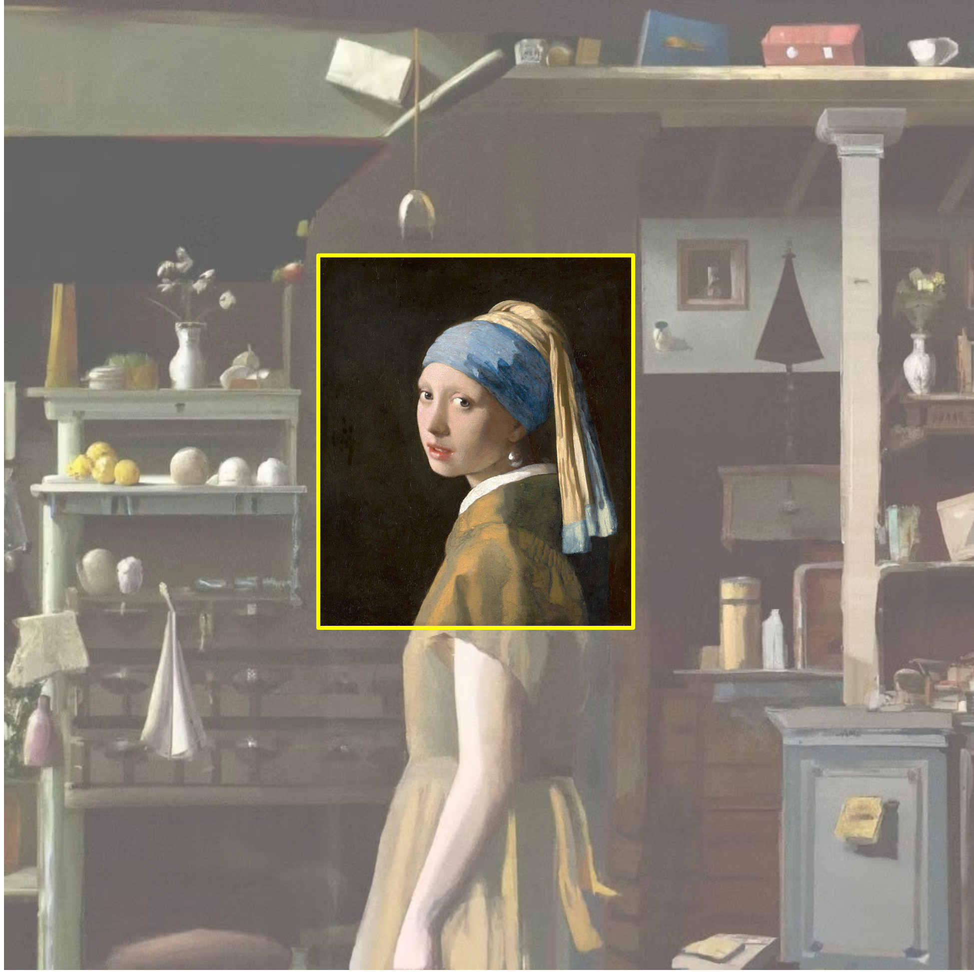


Image by openai.org

Welcome

This is MMI714
“Generative Models for Multimedia”

The course covers all fundamental concepts related to generating media, including latent spaces, latent codes, and encoding.

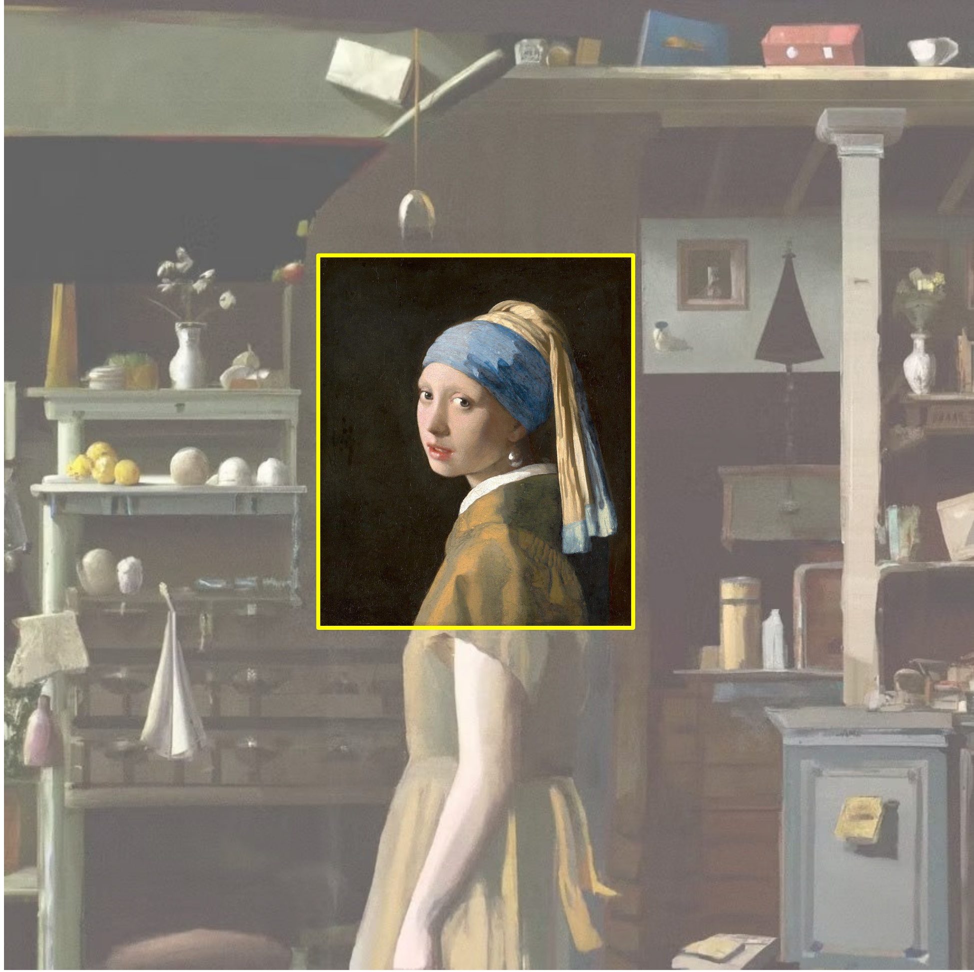


Image by openai.org

Welcome

This is MMI714
"Generative Models for Multimedia"

Throughout the course, students will have access to a wide range of resources, including lectures, readings, and hands-on projects. In addition, a thorough review of recent state-of-the-art studies in the field will be provided each year to ensure students are up to date with the latest advances.

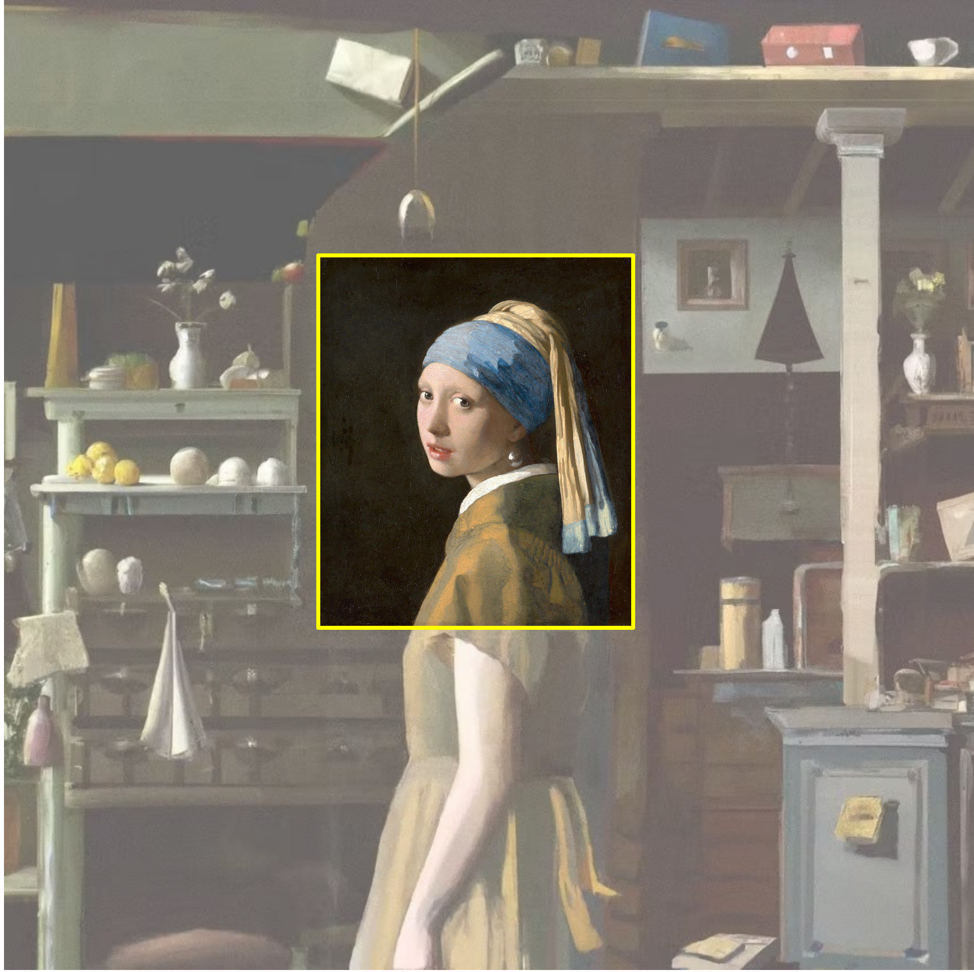


Image by openai.org

Welcome

This is MMI714
"Generative Models for Multimedia"

By the end of the course, students will have gained the skills and knowledge necessary to tackle real-world generative modeling challenges and become proficient practitioners in this exciting field.

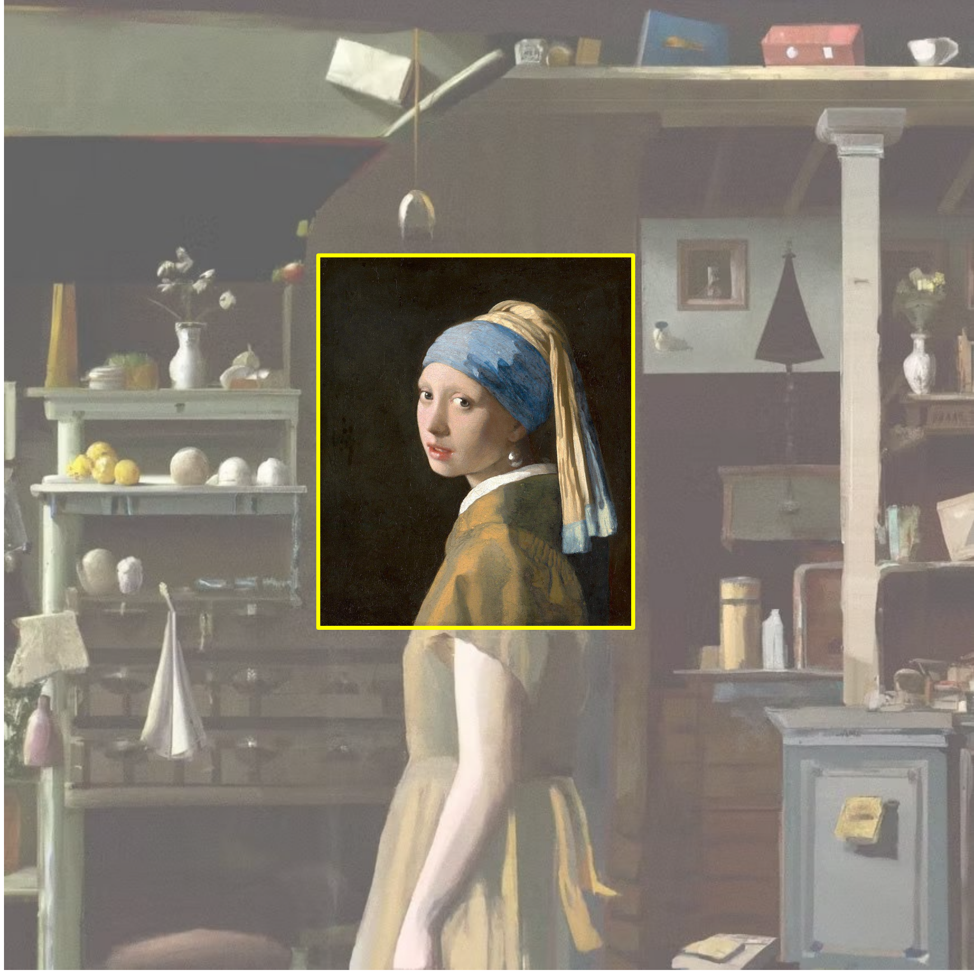


Image by openai.org

Who can take this course?

- This is an advanced deep learning course and requires
  - “an introductory level understanding of deep learning”
- The students should have taken **<u>any one of the courses</u>** below:
  (with at least a CB letter grade)
  - (@metu) MMI727 DEEP LEARNING: METHODS AND APPLICATIONS
  - (@metu) FOUNDATIONS OF DEEP LEARNING
  - (@metu) NEUROCOMPUTERS AND DEEP LEARNING
  - (@metu) DEEP LEARNING
  - Or a similar introductory level deep learning course with the consent of the instructor

Who should take this course?

- In addition,
    - Basic programming skills, most preferably in Python, are required, and students need to know the fundamentals of deep learning.
    - Other programming platforms (such as MATLAB, C etc) are not recommended but can be chosen at your own risk.
    - A basic understanding of probability theory is also a major requirement.

Outline

- Introduction to course,
- Mathematical Background & Fundamental Concepts
- Auto-Regression
- Dimensionality Reduction & Latents Spaces
- AutoEncoders, Variational Inference, VAE
- Flows and Normalizing Flows
- Diffusion Models
- GANs, Generator vs Discriminator, Training GANs, mode collapse
- Evaluating GANs, Sampling GANs, Inverting GANs, Conditioning GANs
- GAN architectures

Grading

- Project 40%
- Final 30%
- Midterm 20%
- Homework 10%

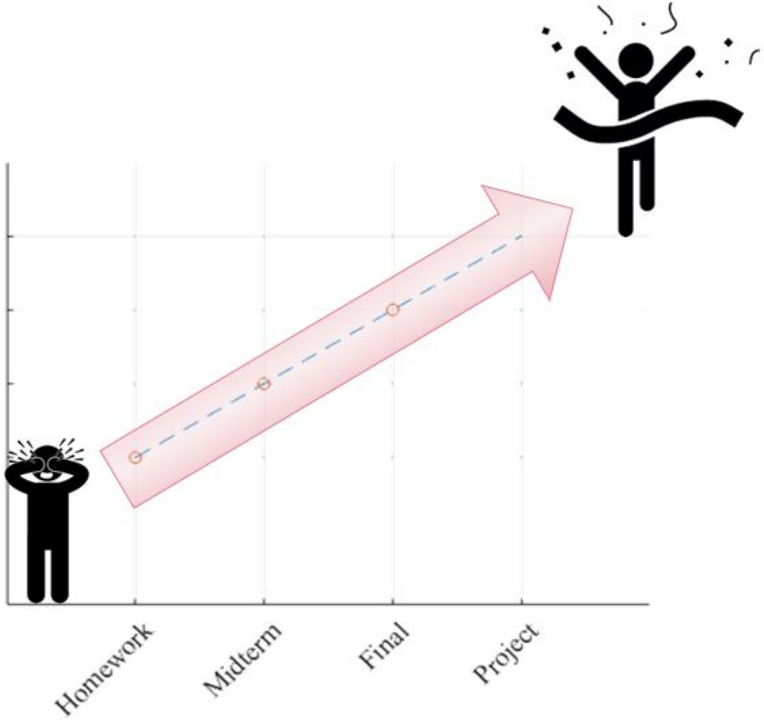


figure by Dr. Akagündüz

Nominated the for the
“most stupid educational figure ever existed awards” (MSSFEEA)

The Exams

Time-limited take-home final (midterm classical)

I will kindly ask:

- for a complete evening of yours,
- usually around 6 hours,
- uploadable to ODTUClass,
- both theoretical and practical questions,
- and to complete a time-limited take-home midterm and final.

The Project

Three main components:

1. Project Proposal (due: early December) **10%** (of the total project grade)
2. Project Presentations (to be held on week 13&14, in class) **40%** (of the total project grade)
3. Project Reports (due: the last day of the finals) **50%** (of the total project grade)
4. **and a possible 10% bonus** (for the total project grade)

The Project (2024)

Three main components:

1. Project Proposal (due: early December) **10%** (of the project grade)

   (any format accepted, to be uploaded to ODTUClass):

   Abstract: Max 250 words describing your planned study, problem definition, and the expected (slight) contribution the study is to make.

   Dataset: This can be provided: The AFAD Dataset!

   The Model: The architecture to be implemented, with additional details (if any).

   The Code: The repository (GitHub etc) to base (if any). If project to be constructed from scratch, the reason why.

   Experiment Plan: The details of experiments to run (nature of comparisons if any, number of runs etc.)

The Project

Three main components:

2. Project Presentations (to be held on week 13&14, in class) **40%** (of the project grade)

   a 15 minute presentation describing:

   - the problem definition,
   - the data,
   - Your model,
   - Your experiment plan, and what you are willing to achieve
   - Your possible contribution
   - and preliminary results (if any).

   The audience is expected to contribute with at least one question to be graded for the **2%** (of the project grade)

The Project

Three main components:

3. Project Reports (due: the last day of the finals) **50%** (of the project grade)

   (to be uploaded to ODTUClass)

   Following IEEE's conference template, a double column max 8 page report, prepared in Latex (preferably) with a clear abstract, problem definition, literature survey and the contribution, methods, the entire results with proper demonstrations and a conclusion.

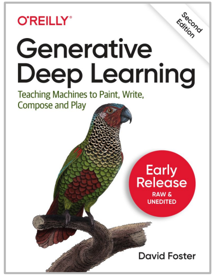
   Depending on the level of contribution of your study, we may consider applying for a conference. **This is the 10% bonus** (for the project grade).

Sources, Book, how-to...

- Sources:
  - Internet! Always, always and always Google, YouTube it, ChatGPT it, Bard it!
  - Some websites always have fantastic stuff on the state-of-the art of AI.

Medium | towards data science | GeeksforGeeks

  - Coursera «Generative Models» course is a must-see! There are others as well!
  - We have a text book: "Generative Deep Learning" by David Foster



Discriminative vs **Generation** Models

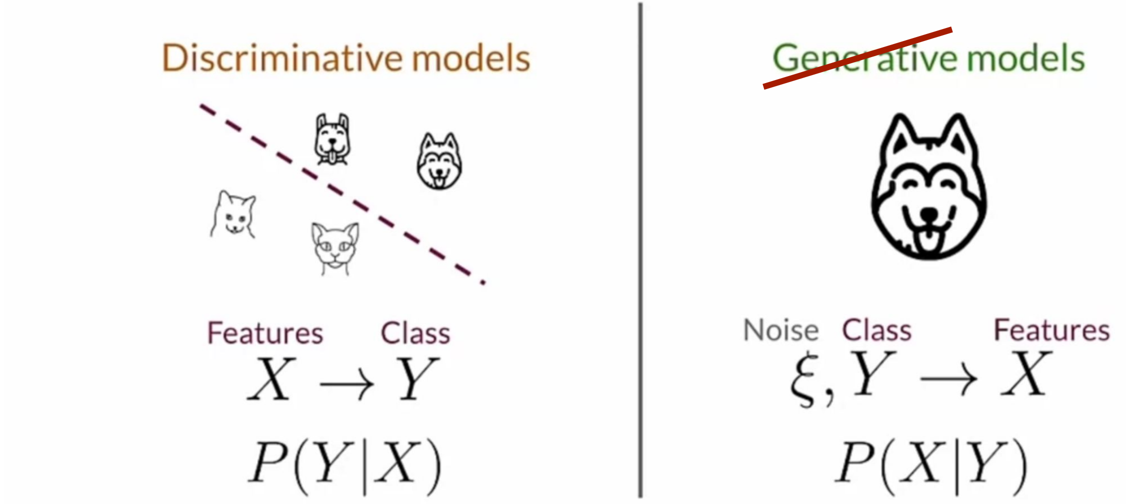
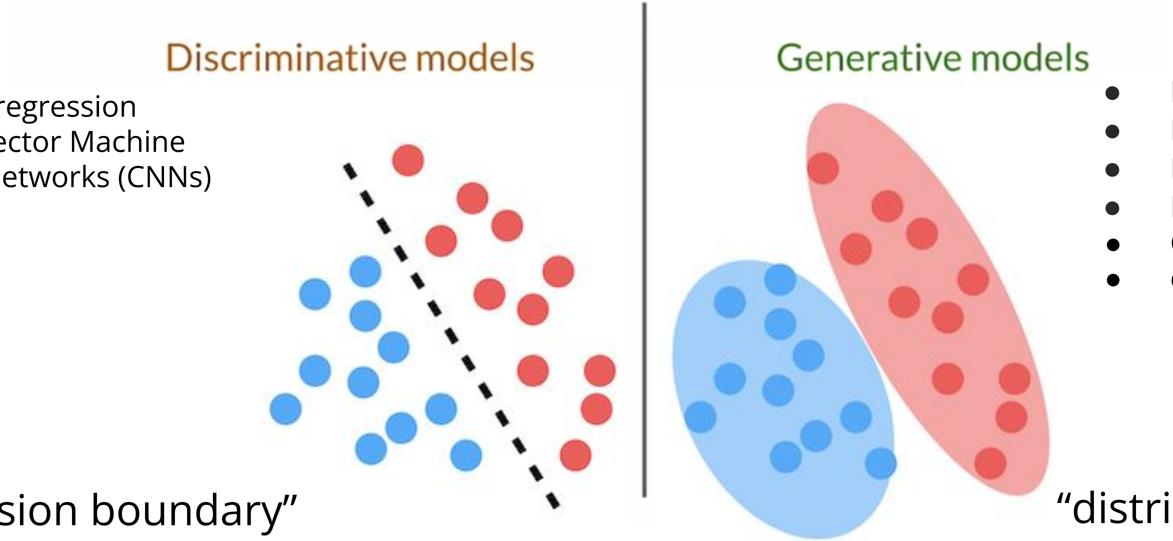


figure by coursera.org

Discriminative vs Generative Models

- Logistic regression
- Scalar Vector Machine
- Neural networks (CNNs)
- etc



- Naïve Bayes
- Bayesian networks
- Markov random fields
- Hidden Markov Models (HMM)
- Gaussian Mixture Models (GMM)
- etc

"decision boundary"

"distribution"

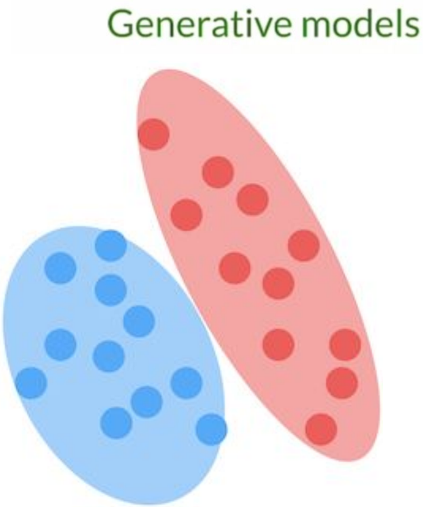
Before the age of GANs, conventional ML scrutinised classifiers into these two categories.

Generative Classifier

A "generative classifier" is a statistical model of the joint probability distribution:

$$P(X,Y)$$

- on given observable variable $\mathbf{X}$
  - Observation $\stackrel{\text{def}}{=}$ Feature
- and target variable $\mathbf{Y}$



- Naïve Bayes
- Bayesian networks
- Markov random fields
- Hidden Markov Models (HMM)
- Gaussian Mixture Models (GMM)
- etc

METU

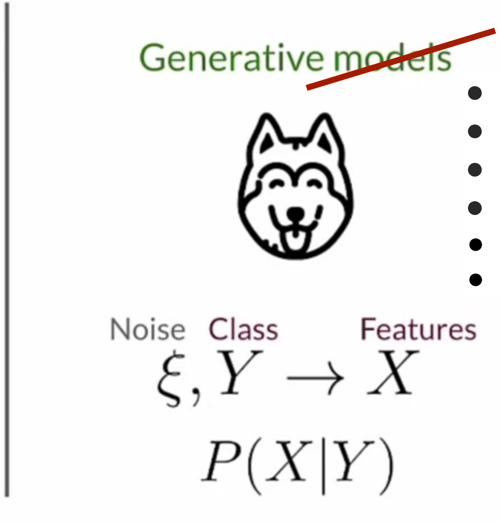
Generative Classifier (vs Generation)

A “generative classifier” is a statistical model of the joint probability distribution:

$$P(X,Y)$$

- on given observable variable **X**
  - Observation $\stackrel{\text{def}}{=}$ Feature
- and target variable **Y**

“When a new observation is fed to a generative classifier, it tries to predict which class would have most likely generated the given observation.”



- Naïve Bayes
- Bayesian networks
- Markov random fields
- Hidden Markov Models (HMM)
- Gaussian Mixture Models (GMM)
- etc

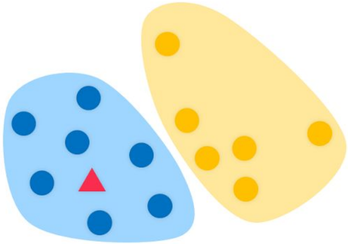


figure by coursera.org

Classification vs Generation

Discriminative models (or classifiers) are machine learning models that learn to classify input data into different categories based on a set of features.

These models learn to distinguish between different classes of data **without necessarily modeling the underlying probability distribution of the data**.

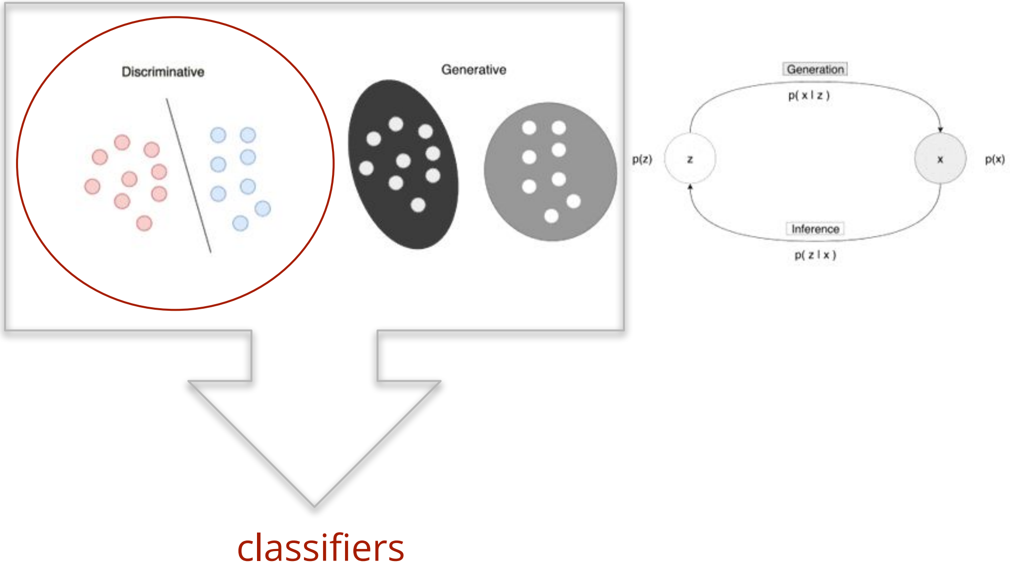


figure by coursera.org

Classification vs Generation

Generative classifiers are models that **learn to model the probability distribution of the input data for each class separately, and then use Bayes' rule to calculate the posterior probability of each class given the input**.

These models can be used to classify new data points into different categories based on their likelihood under each class distribution.

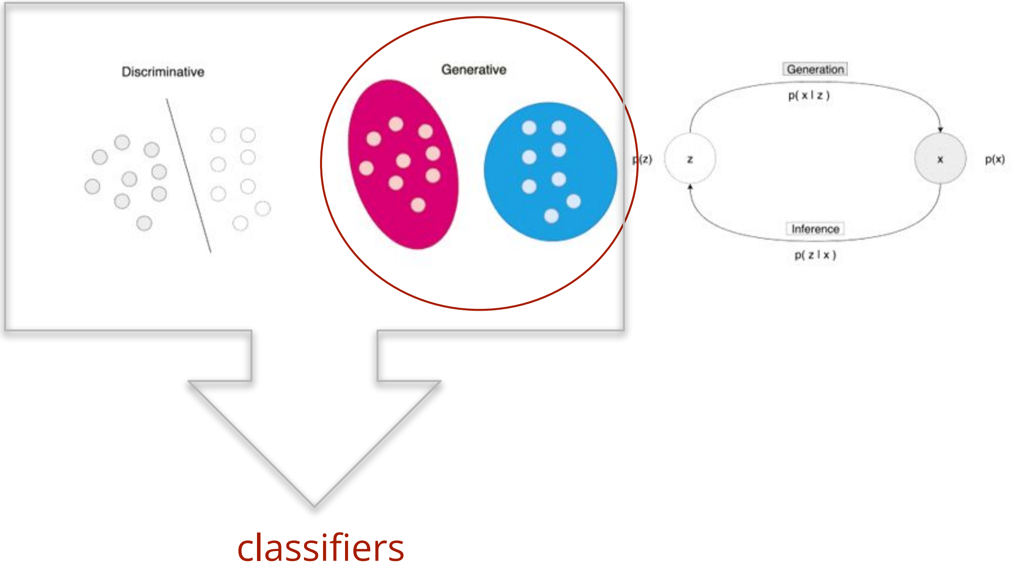


figure by coursera.org

Classification vs Generation

Generative (or generation) models are **machine learning models that learn to model the probability distribution of the input data.**

These models can be used to generate new samples of data that resemble the original data distribution.

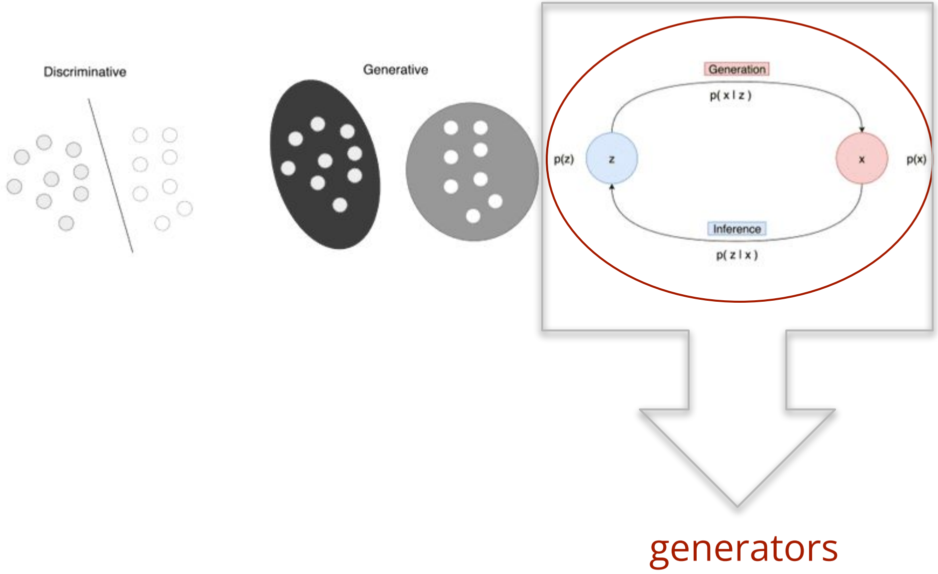


figure by coursera.org

Generation

- $p(z)$: a noise distribution (i.e. latent space)
- $p(x|z)$: is the generation process, i.e. the generation of a new sample x, given a noise vector z
- $p(x)$: is the data/observation distribution.
- $p(z|x)$: is the inference process, i.e. extracting the noise vector z that would generate the sample x.

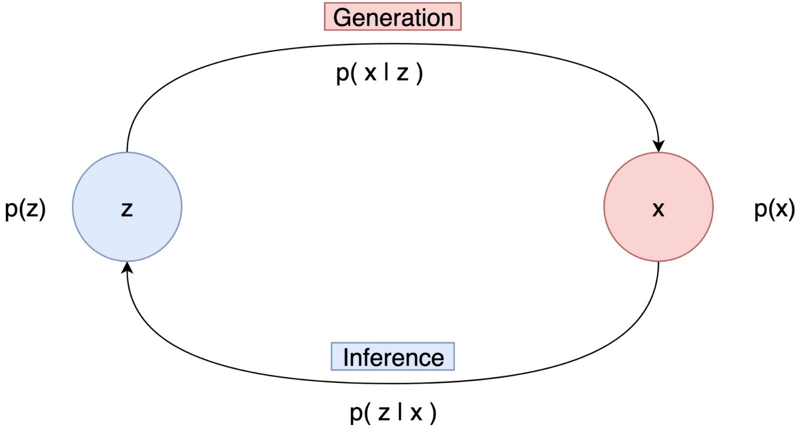


figure by coursera.org

Generation (explicit vs implicit)

Generative models can be divided into two main categories: **explicit** and **implicit**, according to how they **“model the probability distribution of the data”**

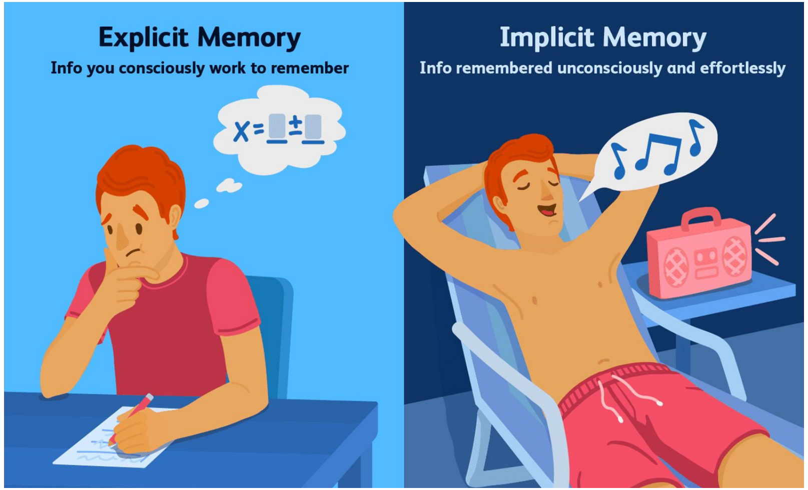


figure by verywellmind.com

Explicit Density Models

These models directly estimate the probability distribution of the data, either through a likelihood function or a probability density function.

In explicit models, the data distribution is modeled directly, typically using a probability density function or a probability mass function. These models are also known as density-based or likelihood-based models.

Examples of explicit density models include:

- Gaussian Mixture Models (GMM)
- Autoregressive Models (e.g., PixelCNN)
- Variational Autoencoders (VAE)
- Diffusion Models

Implicit Density Models

In implicit models, the data distribution is not modeled directly. Instead, these models learn a mapping from a simple noise distribution to the data distribution.

Because the data distribution is not modeled explicitly, these models are also known as density-free or likelihood-free models.

Examples of purely “implicit” density models include:

- Variants of GANs

btw we will also talk about inverting GANs, which is an effort to make them explicit.

Likelihood?

In the context of generative models, the likelihood refers to **the probability of observing a given set of data points under the assumed probability distribution of the generative model.**

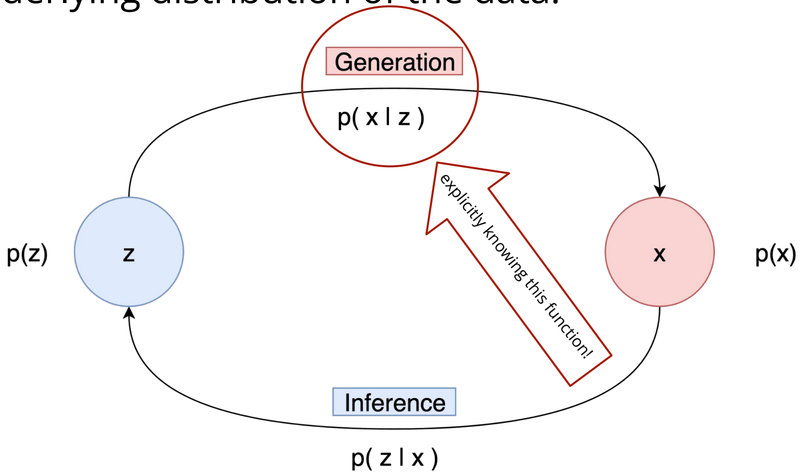
In other words, the likelihood measures how well the generative model can explain the observed data.

- *For example, let's say we have a dataset of images and we want to train a generative model to produce new images that are similar to those in the dataset.*
- *We might choose a probabilistic model that assumes the data is generated from a particular distribution, such as a Gaussian or Bernoulli distribution.*
- *We can then calculate the likelihood of the observed images under this distribution, which tells us how well the model is able to explain the data.*

Likelihood?

The likelihood is often used as the objective function for training *explicit* generative models, as it provides a measure of how well the model is able to fit the data.

Specifically, the goal of training a generative model is typically to maximize the likelihood of the observed data under the model, which corresponds to finding the parameters of the model that best capture the underlying distribution of the data.



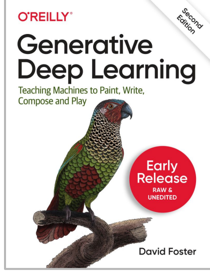
Generation (explicit vs implicit)

All methods and both explicit and implicit approaches have their own strengths and weaknesses.

Explicit models are typically easier to train and can model complex distributions accurately. However, they may suffer from slow generation times or may not be able to generate high-dimensional data such as images efficiently (which is not always true, for example: the diffusion model that the DALL-E 2 is based on).

On the other hand, implicit models can generate high-quality samples quickly and efficiently, but may struggle to model complex distributions accurately and suffer from mode collapse, where the generator produces only a limited range of samples.

Generative Model Taxonomy



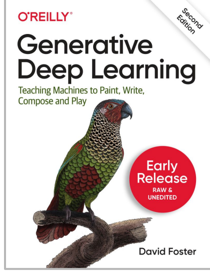
page 18

Generative Model Taxonomy

Whilst all types of generative model ultimately aim to solve the same task, they all take slightly different approaches to modeling the density function $p_\theta(\mathbf{x})$. Broadly speaking, there are three possible approaches.

1. Explicitly model the density function, but constrain the model in some way, so that the density function is tractable (i.e. it can be calculated).
2. Explicitly model a tractable approximation of the density function
3. Implicitly model the density function, through a stochastic process that directly generates data.

Generative Model Taxonomy



page 19

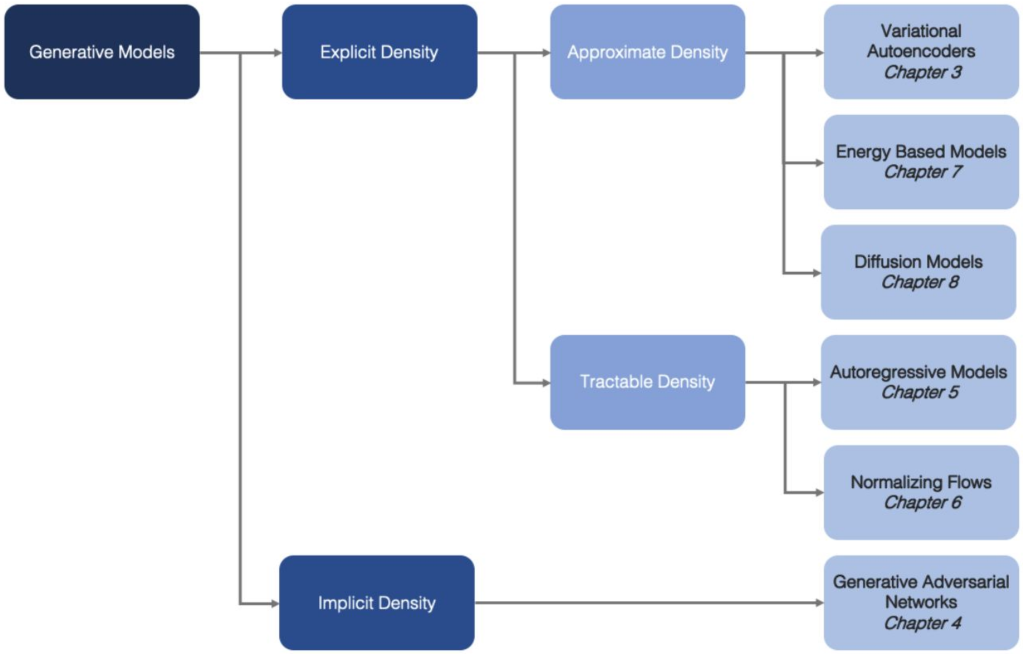
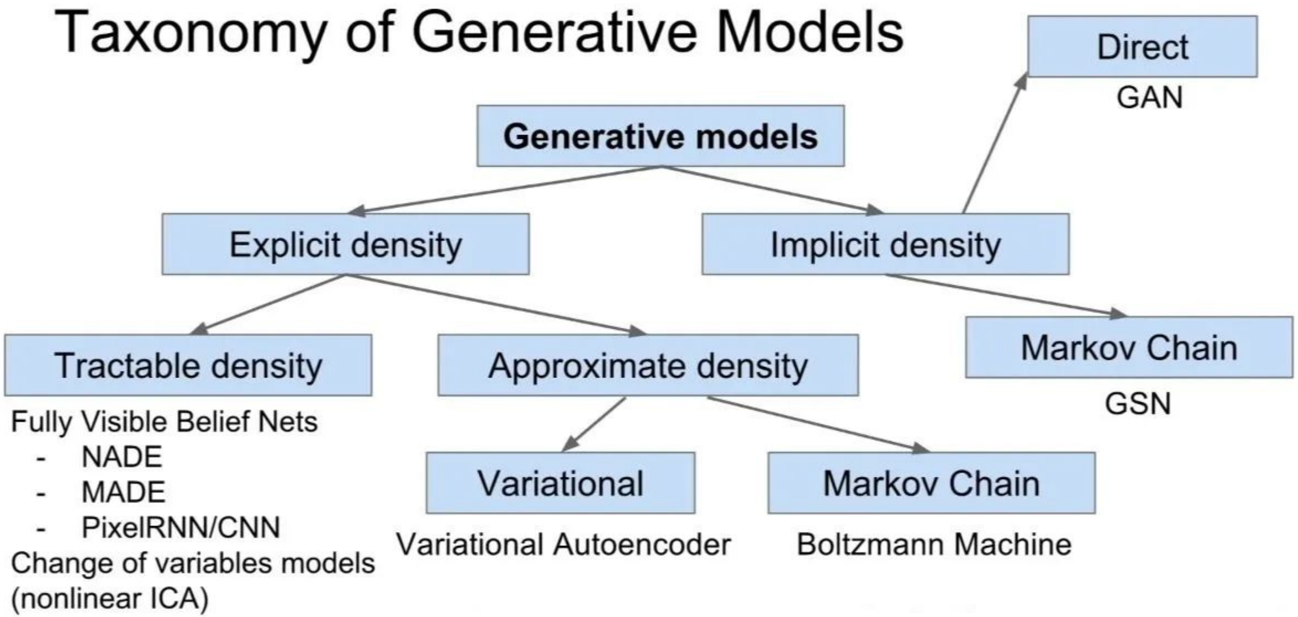


Figure 1-10. A taxonomy of generative modelling approaches

Generative Model Taxonomy



https://medium.com/mlearning-ai/generative-modeling-7a88e847f62e

What to do until next week?

You may

- Go over your probability notes, remember what likelihood, Bayes Theorem, etc were…
- Go over the projects pages on ODTUClass, come up with ideas

What will we do next week?

- Starting with next week...
  - *Sampling, evaluation and training/generating distributions, Bayes theorem, likelihood (variational bound on likelihood, marginal likelihood), Gaussian distributions, modality, complexity, expectation maximization, distribution distances, divergence, KL Divergence, Jensen-Shannon Divergence, etc*
- Following weeks
  - *Time Series, Autoregressive Models, Deep Autoregressive networks, Recurrent Neural Networks for Generation*

Additional Reading & References

- https://www.coursera.org/lecture/build-basic-generative-adversarial-networks-gans/generative-models-4UNci
- https://medium.com/@mlengineer/generative-and-discriminative-models-af5637a66a3
- https://www.youtube.com/watch?v=cmYQNhv5xUw
- https://towardsdatascience.com/a-generative-approach-to-classification-17a0b5876729
- https://medium.com/mlearning-ai/generative-modeling-7a88e847f62e