# ECE 8803-GGDL Homework 4

Arati Ganesh

November 27, 2023

1. Score-based Diffusion Models (50 points)

   b. Part 1: Score and diffusion.
   **i)** Run the cell illustrating the diffusion process. What role does lambda play in this type of process?

   **Answer:**

   In the 1D diffusion process, Lambda ($\lambda$) acts like a diffusion coefficient. Specifically:

   - **Amplification of Noise:** $\lambda$ acts as an exponent in the noise scale function $g(t) = \lambda^t$. A larger $\lambda$ means more noise is added as time progresses, leading to faster and more extensive spreading of the data points.
   - **Stability vs. Chaos:** For $\lambda > 1$, the process becomes more chaotic with time, as noise increases rapidly. For $\lambda < 1$, the process is more stable, with less noise added over time.

   In essence, $\lambda$ controls how quickly and extensively the original data becomes obscured or dispersed due to the added noise.

   ---

   **ii)** Run the cells corresponding to the score for Gaussian Mixtures exercise. What does the direction and magnitude of the score tell you? For a multi-modal distribution, how does the score of the individual mode relate to the overall score?

   **Answer:**

   - **Direction of the Score:** Indicates where the probability density increases most sharply, typically pointing towards the nearest mode (peak) of the nearest Gaussian component in the mixture.
   - **Magnitude of the Score:** Reflects the steepness of the increase in probability density. A larger magnitude suggests a steeper increase, often occurring near the peaks of modes.
   - **Score in Multi-Modal Distributions:** In a distribution with multiple modes, the overall score at any point is a combination of the influences from all modes. It's mainly dominated by the nearest mode(s). The overall score is a weighted sum of the gradients (scores) of individual modes, where weights correspond to the relative strength of each mode at that point.

   ---

   **iii)** Run the cells corresponding to the reversal of the diffusion process, implementing the required functions. How well does the reverse diffusion capture the original distribution?

   **Answer:**

   In evaluating the reverse diffusion process in a diffusion-based model, it's found that while it does a decent job of approximating the original distribution, there are noticeable discrepancies. The reverse process, designed to reconstruct the original data from a noised version, manages to capture the core

features and patterns of the original dataset. However, it does not achieve a perfect overlap, indicating some loss of fidelity or introduction of artifacts during the noise-reduction and reconstruction phases.

---

c. Part 2: Denoising. Implement the denoising score matching objective and test your implemen- tation by running the remaining cells in the section. Recall your interpretation of the score for multi-modal distribution. How does that connect to the denoising objective? In principle, is there a way to optimize the score-matching objective to 0? Explain.

**Answer:**

In denoising score matching, we add Gaussian noise to data and train a model to predict the score, or the gradient of the log probability, of this noisy data. This technique is crucial for multi-modal distributions, as the added noise helps the model explore low-density regions between modes, enhancing its understanding of the overall distribution.
While theoretically possible, optimizing the score-matching objective to zero is practically challenging due to model capacity limits and the complexity of real-world data.

---

d. Part 3: Diffusion on MNIST.

**i)** Implement the variance of the marginal distribution and the diffusion coefficient, and run the cells defining the network architecture. How is time modulation implemented in this model? How might the final normalization step help or harm the score learning?

**Answer:**

The Gaussian Fourier Projection is used to encode the concept of time. This is vital because, in the reverse process of the diffusion model, the way the model reconstructs the image from noise depends heavily on the specific time step of the process. By encoding time steps using sine and cosine transformations, the Gaussian Fourier Projection provides the model with a high-dimensional representation of time. This representation captures the cyclical nature of the process, which is essential for the model to understand how to progressively denoise the image.

The final normalization step in the network is crucial for maintaining a consistent output scale, which aids in stabilizing the training process and ensuring effective learning. However, if overdone, this normalization might limit the network's ability to capture the full range of score dynamics, particularly in regions with subtle variations, potentially hindering the model's performance in accurately learning the score function.

---

**ii)** Implement the loss function, train the model and visualize a few samples. Are you able to notice that the data is generated from a noise distribution?

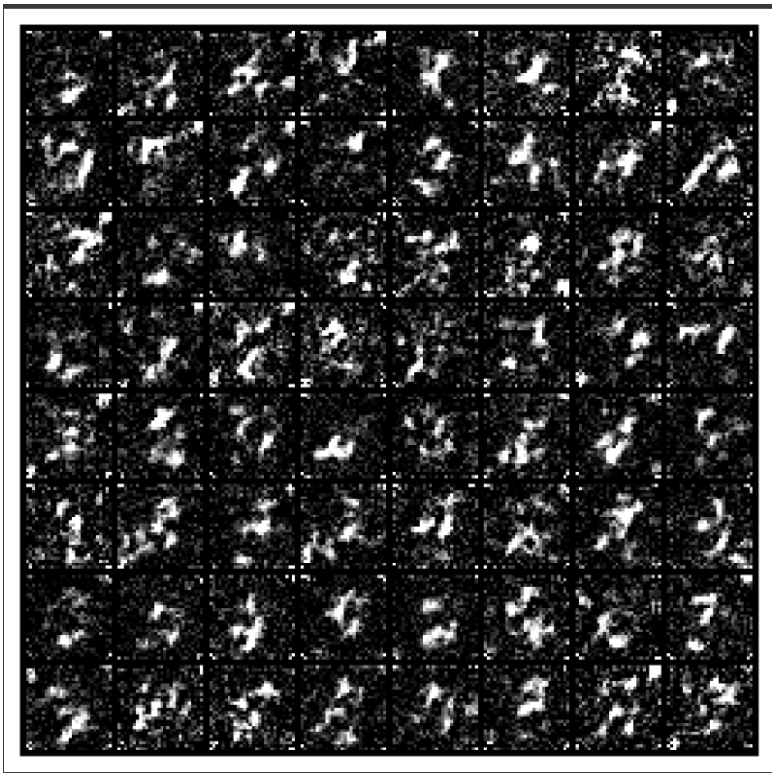

Figure 1: Generated Digits

---

Problem 2: Steerable CNNs (40 points)

(a) **Part 1: Representation Theory & Harmonic Analysis**

i. What do you think happens when irreps have partially or completely redundant entries?
**Ans)** If irreps of a group have partially or completely redundant entries, it implies a loss of independence among the vectors in the representation. In the context of SO(2), this redundancy can manifest in multiple representations encoding the same rotational states, thereby losing the uniqueness and distinctness that ideally characterizes irreps.

ii. How does this affect the resulting basis they constitute?
**Ans)** The basis constituted by these irreps suffers as the redundancy compromises the ability to span the representation space effectively. Ideally, each irrep should correspond to a unique aspect of the group's structure. Redundancy dilutes this uniqueness, leading to an over-representation of certain features while potentially under-representing others.

iii. What happens to the corresponding regular representation?
**Ans)** In a regular representation, each group element is represented by a matrix, with the group operation corresponding to matrix multiplication. When irreps are redundant, the regular representation can become inefficient, as it might include multiple equivalent matrices. This not only leads to computational inefficiencies but also reduces the clarity and utility of the representation in capturing the essence of the group's structure.

---

(b) **Part 2: Steerable CNNs**

i. Does transforming an image with the group $G = C_4$ require any interpolation? Why or why not?
**Ans)** The group C4 represents the cyclic group of 90-degree rotations (i.e., 0, 90, 180, 270 degrees). When transforming an image with C4, interpolation is typically not required. This is because the rotations involved are multiples of 90 degrees, and for standard pixel grids, such rotations map pixels to other pixel locations without the need for intermediate values. This contrasts with rotations by non-multiples of 90 degrees, where interpolation would be necessary to account for pixels that do not align perfectly with the original grid.

ii. Is the model equivariant to $SO(2)$ model perfectly equivariant? Why is this an expected behaviour?
**Ans)** No the model is not perfectly equivariant. This is due to the discrete nature of the image grid and he limited number of sampled orientations.

(c) **Part 3: Build and Train Steerable CNNs**

i. Is perfect invariance to $SO(2)$ achievable by any of the trained models?
**Ans)** No perfect equivariance is not acheieved by any of the models. This is because of the discrete nature of the image grid and the limited number of sampled orientations

ii. Which model is more stable over the rotations of the test set?
**Ans)** Among the models trained for rotation invariance, those specifically designed to handle SO(2) rotations are generally more stable over continuous rotations of the test set than C4. This is because they are designed to be sensitive to a wider range of rotational angles, not just multiples of 90 degrees as in C4 symmetry.

iii. Are both models perfectly equivariant to rotations by multiples of $\pi/2$? Explain.
**Ans)** Models designed for C4 symmetry are typically perfectly equivariant to rotations by multiples of pi/2. For models designed for SO(2) invariance, perfect equivariance to rotations by pi/2 is not always guaranteed. These models are optimized for a wide range of rotations, and although they generally handle /2 rotations effectively, the approximations required for continuous rotations can prevent perfect equivariance.

---

Problem 3: Sufficient condition for equivariance on graphs (10 points). Prove the con- jecture regarding the equivariance of GNNs in slide 13 of lecture 21

Ans)

Problem 3.

To ensure equivariance, it is sufficient if $\phi$ does not depend on the order of nodes in $x_{N_i}$ (i.e if it is permutation invariant)

To prove :-

consider consistent permutations of the shift operator $\hat{S} = P^T S P$ and input signal $\hat{x} = P^T x$. Then -

$$\phi(\hat{x}, \hat{S}, H) = P^T \phi(x, S; H)$$

Proof :- GNN Layer $l$ recursion on signal $x_{l-1}$ and shift $S \Rightarrow$

$$x_l = \sigma\left[\sum_{k=0}^{k-1} h_{lk} S^k x_{l-1}\right] = \sigma\left[H_l(S) x_{l-1}\right]$$

GNN Layer $l$ recursion on signal $\hat{x}_{l-1}$ and shift $\hat{S} \Rightarrow$

$$\hat{x}_l = \sigma\left[\sum_{k=0}^{k-1} h_{lk} \hat{S}^k \hat{x}_{l-1}\right] = \sigma\left[H_l(\hat{S})\, \hat{x}_{l-1}\right]$$

→ Assume Layer $l$ inputs satisfy $\hat{x}_{l-1} = P^T x_{l-1}$. Filters are equivariant. Linearity is pointwise.

$$\hat{x}_l = \sigma\left[H_l(\hat{S})\, \hat{x}_{l-1}\right] = \sigma\left[P^T H_l(S)\, x_{l-1}\right] = P^T \sigma\left[H_l(S) x_{l-1}\right]$$

$$= P^T x_l$$

→ This is an induction step. At layer 1 we have $\hat{x} = P^T x$

Induction is complete.

Problem 4: Different realization of graph neural networks (10 points). Do the exercise outlined in slide 43 of lecture 21.

Ans)

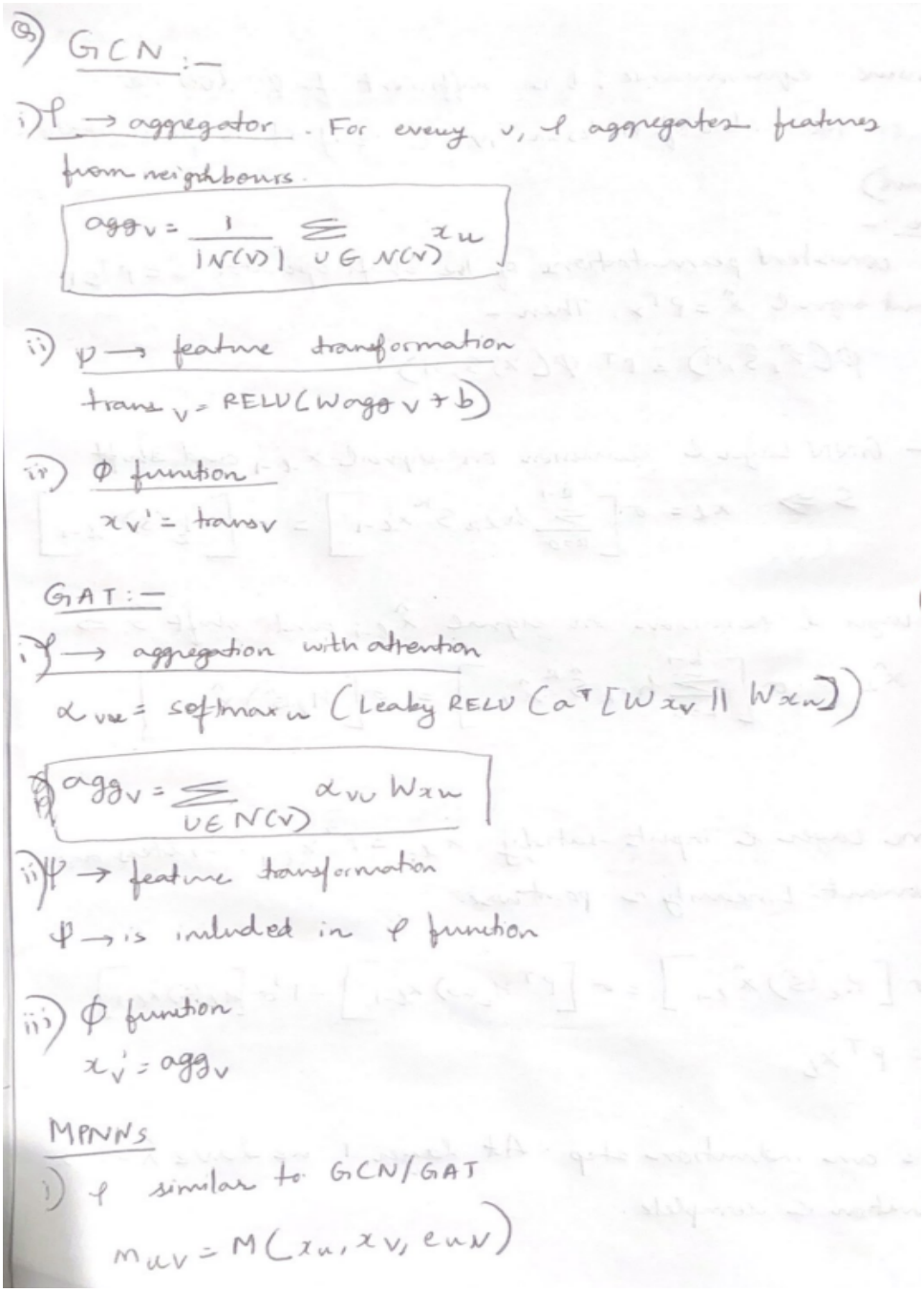


Figure 3:

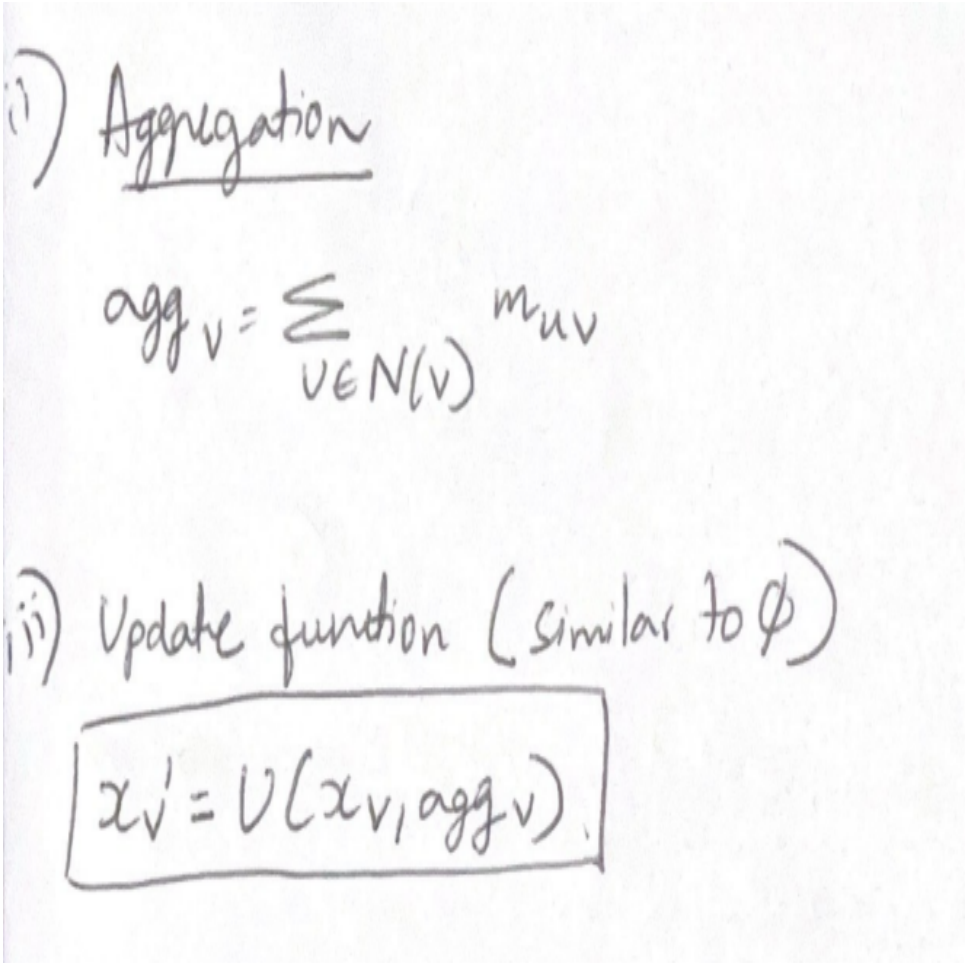
(i) Aggregation
$agg_v = \sum_{u \in N(v)} m_{uv}$
(ii) Update function (similar to $\phi$)
$x_v' = U(x_v, agg_v)$

Figure 4: