**Reflectie-verslag: Implementatie van een Denoising Diffusion Probalistic Models**

**Studenten:**

Julia Boschman

Daan Eising

**Studentnummers:**

22074845

22162038

**Datum:**

23 maart 2025

1. **Inleiding**

Voor het vak Generative AI hebben wij, Julia Boschman en Daan Eising, ervoor gekozen om een Denoising Diffusion Probabilistic Model te implementeren aan de hand van een dataset met Barokstijl schilderijen, gebaseerd op het paper “Denoising Diffusion Probabilistic Models”. Het doel was om het model volledig “from scratch” te programmeren. Hierbij mochten wij geen externe Python-packages gebruiken om het model uit te voeren, behalve NumPy, Pandas, math en TensorFlow zonder Keras. In dit verslag benoemen wij de technische uitdagingen en leermomenten die wij tijdens het maken van het portfolio zijn tegengekomen.

**Problemen en oplossingen**

Tijdens het maken van het portfolio zijn we tegen meerdere problemen aangelopen. Hieronder benoemen we deze problemen en de uiteindelijke oplossingen die we hebben doorgevoerd om deze zo goed mogelijk te verhelpen.

**Starten**

Het eerste probleem waar wij als duo tegenaan liepen, was het starten van het portfolio. Na de eerste les van het vak hebben wij samen een planning opgesteld om ervoor te zorgen dat wij het portfolio vroegtijdig zouden afronden. Echter, de vooraf opgestelde planning bleek niet realistisch en was daardoor lastig om aan te houden. De voornaamste reden hiervoor was het bijbehorende paper; het lezen en begrijpen van het paper vormde voor ons een grote drempel. Daarnaast speelde er nog een andere uitdaging: de nieuwe stof en de grote hoeveelheid huiswerk die losstond van het portfolio. Wij waren deze belasting niet gewend en hadden tijd nodig om ons hierop aan te passen. Deze combinatie heeft ervoor gezorgd dat wij het portfolio uit het zicht zijn verloren.

**Oplossing voor het opstart probleem.**

De oplossing voor dit probleem kwam tot stand in lesweek 4. Na enkele aanpassingen aan het vak kregen wij meer ademruimte en kwamen we tot de conclusie dat we nu stappen moesten maken met het portfolio. Daarom hebben wij als duo afgesproken om actiever met elkaar te communiceren en duidelijke afspraken te maken. Een van deze afspraken was dat we iedere schooldag na de contacturen samen op een rustige locatie zouden werken aan het portfolio. Dit heeft ervoor gezorgd dat we daadwerkelijk zijn begonnen en grote stappen in de juiste richting hebben gezet.

**Complexiteit paper**

Toen wij uiteindelijk in lesweek 4 met meer focus aan het portfolio begonnen, lukte het ons beiden niet om de paper eigen te maken en daadwerkelijk te begrijpen. In week 1 hadden wij gekozen voor de portfolio-opdracht: Normalizing Flow Model op basis van het paper “Variational Inference with Normalizing Flows”. Na meerdere pogingen is het ons niet gelukt om de code “from scratch” te implementeren.

**Oplossing complexiteit paper**

Door de tegenslag bij het implementeren en begrijpen van het paper was onze motivatie flink gedaald. Uiteindelijk kwamen we tot de conclusie dat het “from scratch” implementeren op basis van dit paper niet zou lukken. Vervolgens hebben we de overige papers bekeken om er één te selecteren voor een tweede poging. De paper “Denoising Diffusion Probabilistic Models” sprak ons hierbij erg aan. In dezelfde periode als het portfolio waren we ons ook aan het voorbereiden op de paperpresentatie over Causal Fusion. Deze paper had veel raakvlakken met de paper over DDPM. Hierdoor begrepen we het paper beter, hadden we meer ideeën voor mogelijke datasets en keerde onze motivatie terug. Uiteindelijk hebben we na deze keuze harder en met meer enthousiasme aan het portfolio gewerkt.

**Wiskundige concepten**

Na het veranderen van paper begrepen we het concept beter. Echter, het paper bevatte geavanceerde wiskunde, wat het lastig maakte om dit om te zetten naar code. Vooral de formules en achterliggende concepten zorgden ervoor dat we veel tijd moesten besteden aan het begrijpen ervan.

**Oplossing moeilijke wiskundige concepten**

Om de wiskunde toch goed om te kunnen zetten, hebben we externe bronnen geraadpleegd om de concepten en formules in kleinere delen op te splitsen en zo beter te begrijpen. De grootste hulp hierbij was ChatGPT. Met deze tool was het mogelijk om alles wat we niet begrepen steeds simpeler te laten uitleggen, totdat we het begrepen. Op deze manier werd het alsnog mogelijk om de informatie uit het paper eigen te maken.

**Trainingstijd**

Toen we eindelijk het model hadden geïmplementeerd, kwam er een nieuw probleem aan het licht. Door het “from scratch” implementeren van het model hadden we niet de gebruikelijke optimalisatietechnieken toegepast die normaal gesproken standaard zijn inbegrepen in Keras en PyTorch. Daarnaast konden we het model niet laten draaien via een GPU; alles verliep via NumPy en TensorFlow (zonder Keras), waarbij alleen de CPU werd benut. Hierdoor was de trainingstijd van het model aanzienlijk langer dan verwacht. Het trainen van één epoch duurde ongeveer 20 minuten. Met deze snelheid zou het trainen van het model te veel tijd in beslag nemen.

**Oplossing trainingstijd**

Om het model alsnog goed te kunnen trainen, hebben we het geoptimaliseerd door vectorisatie toe te passen. Daarnaast hebben we de afbeeldingsgrootte van 128x128 verkleind naar 32x32. Dit zorgde er uiteindelijk voor dat de trainingstijd per epoch werd teruggebracht naar ongeveer 1 minuut. Hierdoor was het mogelijk om het model daadwerkelijk te trainen en efficiënter verschillende parameters te testen en aan te passen.

**Gegenereerde samples**

Na het uitvoeren van meerdere tests en verschillende trainingsloops bleef het model afbeeldingen genereren waarbij alleen de toegevoegde ruis zichtbaar was. Bovendien bleef de bijbehorende ruis erg hoog; tijdens onze beste trainingsloop was de laagste loss die we zagen 0.86. Verder bleven de gemiddelde gradiënten erg hoog. Dit gaf aan dat het model onvoldoende leerde en dat er iets misging bij het reverse-proces.

**Oplossing gegenereerde samples**

Nadat we hadden geconcludeerd dat de fouten van het model voortkwamen uit het feit dat het reverse-proces niet goed werkte, hebben we dit uitgebreid onderzocht en veel geëxperimenteerd. Het is ons echter niet gelukt om de loss te verkleinen en het reverse-proces te verbeteren. Uiteindelijk is het ons dus niet gelukt om dit probleem op te lossen. In de toekomst zullen we proberen een soortgelijk probleem te voorkomen of op te lossen door eerder te beginnen en meer hulp te vragen aan de vakdocent.

**Onderschatting complexiteit en benodigde tijd**

Hoewel de opdracht interessant en erg leerzaam was, bleek de complexiteit en de benodigde tijd uiteindelijk groter dan verwacht. Voor het eerst volledig “from scratch” een model uit een paper bouwen en trainen op nieuwe data was erg uitdagend en tijdrovend.

**Oplossing onderschatting complexiteit en benodigde tijd**

Voor de komende portfolio’s en in de toekomst zullen we meer tijd reserveren voor de opdracht en een realistischere planning maken. Op deze manier willen we voorkomen dat we een niet volledig geïmplementeerd model opleveren.

1. **Nieuwe bevindingen**

Tijdens het implementeren van het Denoising Diffusion Probabilistic Model (DDPM) op een dataset van Barokstijl schilderijen, zijn we tegen diverse uitdagingen en opmerkelijke bevindingen aangelopen die niet expliciet in het paper “Denoising Diffusion Probabilistic Models” worden genoemd.

**Nieuwe Bevindingen**

1. *Effect van datasetresolutie op het trainingsproces*
   * Het oorspronkelijke paper maakt gebruik van 32x32 en 256x256 resoluties. Onze initiële implementatie op 128x128 schilderijen veroorzaakte extreme vertragingen (ongeveer 20 minuten per epoch).
   * Het verkleinen van de afbeeldingsresolutie naar 32x32 verkortte de trainingstijd aanzienlijk tot ongeveer 1 minuut per epoch. Dit toont aan dat het model niet alleen gevoelig is voor dataset kwaliteit, maar ook sterk afhankelijk is van afbeeldingsgrootte. Dit aspect wordt niet besproken in de originele paper.
2. *Optimalisatie via Vectorisatie voor CPU-gebaseerde training*
   * Omdat onze implementatie GPU-gebruik uitsloot (door het verbod op Keras en PyTorch), hebben we vectorisatie toegepast met NumPy en TensorFlow (zonder Keras).
   * Deze optimalisatie verminderde de rekentijd drastisch. Het originele paper behandelt geen optimalisaties voor CPU-gebaseerde implementaties. Onze bevinding toont aan dat deze implementaties aanzienlijk verbeterd kunnen worden door vectorisatie toe te passen.
3. *Uitdagingen bij het Reverse Proces*
   * Het gegenereerde model produceerde voornamelijk ruis met een beste loss van ongeveer 0.86. Dit wijst op een slecht functionerend reverse proces.
   * Het toevoegen van technieken zoals KL-Divergence optimalisatie, Gradient Clipping, Learning Rate Scheduling hebben geen verbetering gebracht aan het model. Sterker nog, de resultaten waren slechter dan voorheen. Daarom hebben wij deze niet geïmplementeerd in de code van het model.
4. *Netwerkarchitectuur en implementatieverschillen*
   * Onze implementatie maakt gebruik van een volledig verbonden netwerk (fully connected layers) met LayerNorm en LeakyReLU activatiefuncties. Het originele paper beschrijft echter een U-Net architectuur met convolutionele lagen en zelf-attentie mechanismen.
     + Deze aanpak hebben wij als eerste wel geprobeerd, maar dit werkte niet. We hebben een poging gedaan om een U-net architectuur toe te voegen aan het model. Dit hebben wij uitgevoerd met de google colab T4 GPU, echter kostte dit te veel GPU-memory en runde we al snel “out-of-memory”. Daarom hebben we ervoor gekozen om toch een andere aanpak te gebruiken, waarbij wij alleen een CPU gebruiken, die de wiskunde gebruikt uit de paper en voor ons toch belovende resultaten laat zien.
   * Het gebruik van volledig verbonden lagen zonder convolutionele lagen vermindert mogelijk het vermogen van het model om ruimtelijke patronen in afbeeldingen te leren. Dit wordt niet besproken in het originele paper.
5. **Conclusie**

Het implementeren van het Denoising Diffusion Probabilistic Model from scratch was een uitdagende en leerzame ervaring. Onze bevindingen laten zien dat het gebruik van een kleinere afbeeldingsresolutie en vectorisatie cruciaal zijn voor het versnellen van CPU-gebaseerde trainingen. Daarnaast hebben we inzicht verkregen in waarom het reverse proces mogelijk niet naar behoren functioneert, wat niet expliciet besproken wordt in de originele paper.

Nieuwe bevindingen laten zien dat het toevoegen van technieken zoals KL-Divergence optimalisatie, gradient clipping, learning rate scheduling niet alleen ineffectief waren, maar zelfs een negatief effect hadden op de prestaties. Daarnaast lijkt het gebruik van volledig verbonden lagen in plaats van een U-Net architectuur een belangrijke beperking te zijn voor ons model.

Voor toekomstige projecten zullen we eerder beginnen en gerichter ondersteuning vragen van de vakdocent. Daarnaast zal het verder onderzoeken van optimalisatiemethoden voor CPU-gebaseerde training en verbeteringen in het reverse proces onze prioriteit zijn.

1. **Bronnen**

* De Vries, J. (2024, 26 januari). *Reflectie-verslag: Implementatie van een score-based model* [Reflectieverslag]. De Haagse Hogeschool. <https://brightspace.hhs.nl/content/enforced/121890-ADSAI-GENAI-24_2024_SPRING_2/voorbeeldreflectie.pdf?ou=121890>
* *ChatGPT - Reflectie Spelfouten Correctie*. (2025). ChatGPT. <https://chatgpt.com/share/67e29af3-bda8-8000-908b-77bb8c91a0c8>