Implementacja modelu dyfuzyjnego do generowania funkcji matematycznych Raport

Głębokie sieci neuronowe w mediach cyfrowych

Kamil Jaworski
Konrad Gieleta

Warszawa 2024

Spis treści

1 Wstęp

Celem projektu była implementacja modelu dyfuzyjnego do generowania funkcji matematycznych - falek Morleta. W tym celu zaimplementowano model wzorując się głównie na implementacji zawartej w [5] i dostosowując ją pod własne zagadnienie. Stworzono własny dataset, który na początku procesu trenowania generuje zestaw danych treningowych jednowymiarowych do nauki modelu. Założeniem było generowanie takich danych przez model, aby widoczne było wyraźne podobieństwo między funkcją oryginalną a funkcją wygenerowaną z całkowitego szumu.

2 Realizacja projektu

Projekt realizowano w kilku etapach, zaczynając od stworzenia datasetu, przez implementację modelu, aż po przeprowadzenie eksperymentów oceniających jakość generowanych funkcji.

2.1 Baseline

W projekcie zaimplementowano podstawowy model dyfuzyjny oraz uproszczone modele porównawcze, aby ocenić ich skuteczność w generowaniu falek Morleta.

Projekt realizowano przy użyciu biblioteki PyTorch oraz PyTorch Lightning, które zapewniły elastyczne środowisko do trenowania modeli głębokiego uczenia. Kluczowym elementem była architektura U-Net, znana ze swojej skuteczności w zadaniach związanych z przetwarzaniem obrazów i sygnałów. Chociaż U-Net jest często stosowany do przetwarzania obrazów, w tym projekcie okazał się nieco przesadnym rozwiązaniem dla analizy sygnałów jednowymiarowych, jednak mimo to sprawdził się dobrze w realizacji zadania.

W pierwszej fazie model trenowano na idealnych, niezaszumionych danych. Proces generowania czystych danych opierał się na funkcji falki Morleta, która została zaimplementowana w Pythonie. Wygenerowane dane były następnie wykorzystywane do trenowania modelu, aby ocenić jego zdolność do wiernego odwzorowania oryginalnych fal.

W kolejnej fazie do danych wprowadzono szum. Był on generowany za pomocą losowych wartości zgodnych z rozkładem normalnym, co symulowało realistyczne zakłócenia w danych. Model uczono radzenia sobie z tym zakłóceniem poprzez propagację w przód i wstecz, zgodnie z teorią modeli dyfuzyjnych. Proces ten polegał na stopniowym dodawaniu szumu do danych wejściowych i późniejszym jego usuwaniu przez model w celu odtworzenia oryginalnego sygnału.

Wynikiem tych działań była seria wizualizacji przedstawiających zarówno zaszumione, jak i odszumione fale, co pozwoliło na obiektywną ocenę skuteczności modelu. Dodatkowo, projekt zawierał eksperymenty z różnymi harmonogramami wartości beta, które kontrolują poziom szumu dodawanego na każdym etapie trenowania.

W ostatnim kroku przeprowadzono testy, które potwierdziły zdolność modelu do efektywnej redukcji szumu i generowania falek Morleta.

2.2 Usprawnienia

W etapie usprawnień nie zrobiliśmy już żadnych zmian w kodzie. Czas przeznaczony na ten etap projektu poświęciliśmy na dobre zrozumienie implementacji, na której się wzorowaliśmy oraz na dokładną analizę wyprowadzeń wzorów matematycznych stosowanych przez nas w projekcie do nauki modelu dyfuzyjnego.

3 Eksperymenty

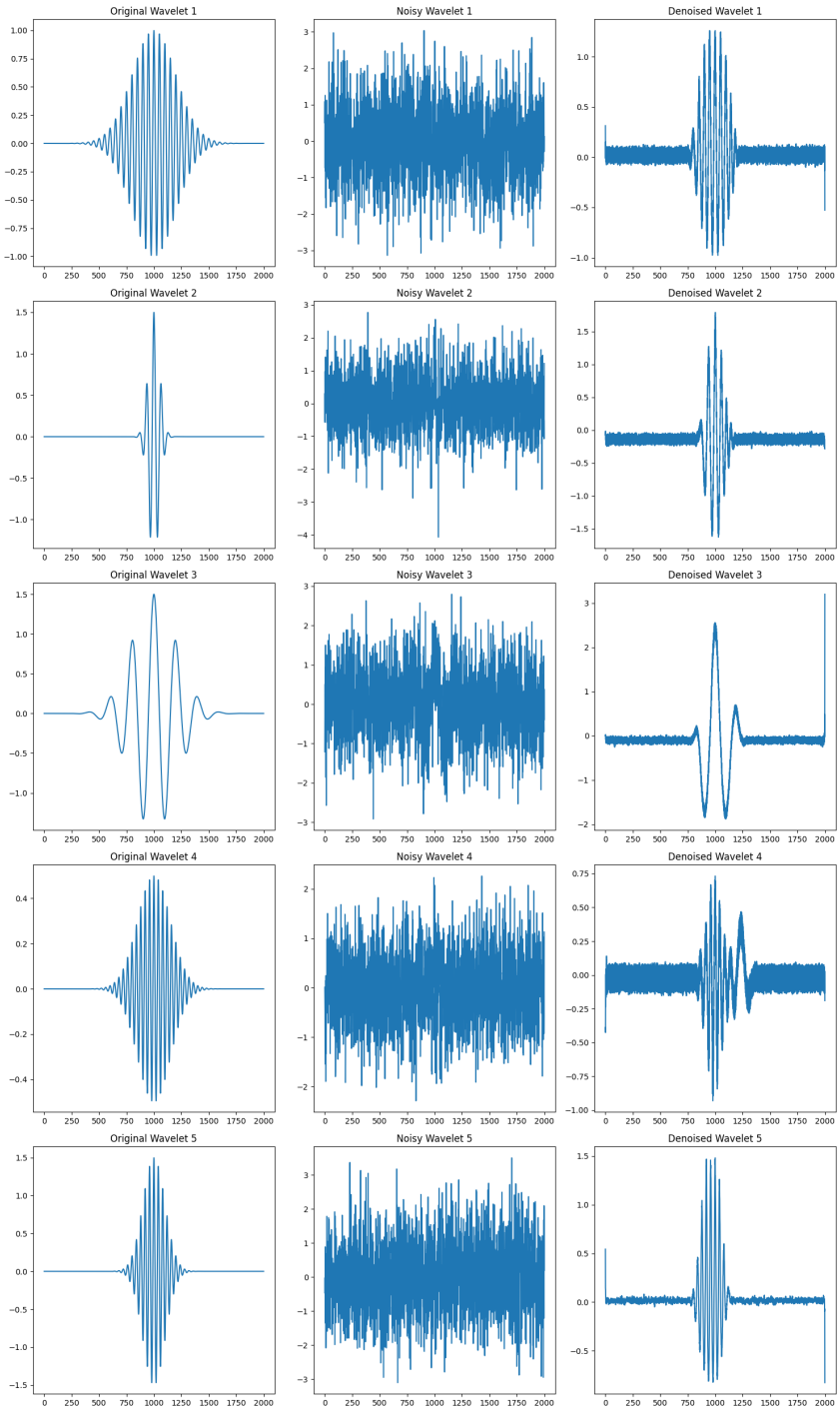
W celu oceny wydajności modelu przeprowadzono eksperymenty, które były skoncentrowane na ocenie jakości generowanych falek Morleta oraz zdolności modelu do redukcji szumu.

3.1 Dodawanie szumu i jego redukcja

W jednym z kluczowych eksperymentów wprowadzono do danych treningowych losowy szum, aby sprawdzić zdolność modelu do jego redukcji i odtwarzania czystych fal. Szum generowano za pomocą rozkładu normalnego, co symulowało realistyczne zakłócenia w danych. Model uczono na zaszumionych danych, aby nauczył się identyfikować i usuwać szum, a następnie odtwarzać oryginalne, czyste fale.

Proces odszumiania polegał na propagacji zaszumionych danych przez model, który na każdym etapie przewidywał i usuwał szum, stopniowo zbliżając się do czystej fali. Model dyfuzyjny przewidywał poprzednią zaszumioną próbkę oraz szum dodany na każdym etapie, a nie bezpośrednio czystą falę. Wyniki eksperymentu pokazały, że model jest w stanie skutecznie zredukować szum i wygenerować fale bliskie oryginałowi.

Przykłady wyników przedstawione są na rysunku 1, który ilustruje proces odszumiania dla kilku różnych falek Morleta. Wizualizacje te pokazują zarówno zaszumione fale, jak i fale odszumione przez model, co pozwala na ocenę efektywności procesu odszumiania.

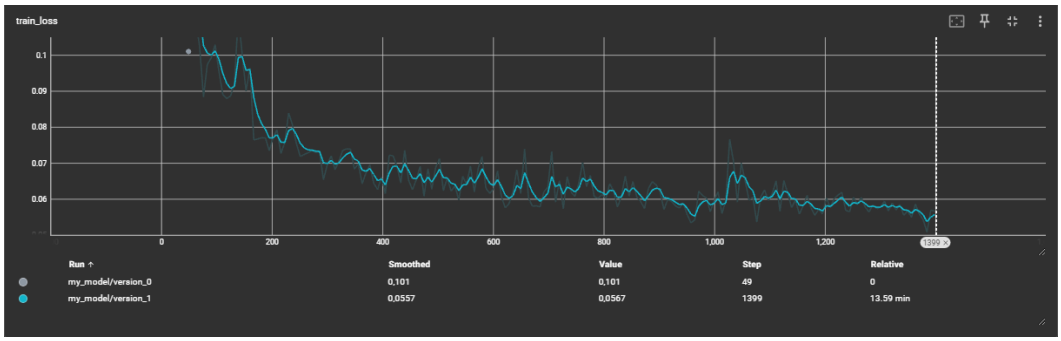


Rysunek 1: Wizualizacja falek Morleta przed i po odszumianiu w różnych krokach czasowych. Lewa kolumna: oryginalne fale; środkowa kolumna: zaszumione fale; prawa kolumna: odszumione fale.

Proces odszumiania polegał na propagacji zaszumionych danych przez model, który na każdym etapie przewidywał poprzednią zaszumioną próbkę oraz szum dodany na każdym etapie, a nie bezpośrednio czystą falę. Wyniki potwierdziły, że model jest w stanie nauczyć się i zastosować skomplikowane operacje na danych, aby

osiągnąć oczekiwane rezultaty. Te eksperymenty dostarczyły cennych informacji na temat wydajności i skuteczności modelu w praktycznych zastosowaniach związanych z przetwarzaniem sygnałów.

Dalsze badania obejmowały także testowanie różnych technik augmentacji danych oraz regularizacji, co pozwoliło na dalsze zwiększenie stabilności modelu i poprawę jego ogólnej wydajności. Dzięki temu możliwe było uzyskanie bardziej precyzyjnych wyników oraz lepszego dopasowania modelu do rzeczywistych danych sygnałowych.



Rysunek 2: Przebieg funkcji straty dla kolejnych kroków treningowych.

4 Podsumowanie

Projekt zakończył się sukcesem, osiągając założone cele. Model dyfuzyjny został skutecznie zaimplementowany i przetestowany, a wyniki eksperymentów potwierdziły jego zdolność do generowania oraz odszumiania falek Morleta. W przyszłości planowane jest rozszerzenie projektu o inne typy funkcji matematycznych oraz eksperymenty z bardziej zaawansowanymi harmonogramami szumu. Dodatkowo, planuje się zastosowanie modelu do bardziej skomplikowanych sygnałów, co pozwoli na ocenę jego ogólnej zdolności do redukcji szumu i generacji sygnałów.

5 Wyprowadzenia wykorzystywanych wzorów

[2][3][4]

5.1 Wyjaśnienie wykorzystywanych symboli i oznaczeń

- $\mathcal{N}$ - Rozkład normalny
- x_t - Próbka w pewnym kroku czasowym t
- x_T - Końcowy, całkowicie zaszumiony obraz po T krokach czasowych
- β - pewien ustalony harmonogram wykorzystywany do odpowiedniego skalowania średniej oraz wariancji. Kontroluje dodawanie szumu w kolejnych krokach w taki sposób, aby nie było ono zbyt szybkie oraz żeby obraz nie był całkowicie zaszumiony dużo szybciej niż w kroku T.
- I - Macierz jednostkowa

5.2 Dodawanie szumów - propagacja w przód

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

gdzie:

- $q(x_t|x_{t-1})$ - obraz zaszumiony w kroku czasowym t znając obraz z kroku $t-1$
- $\sqrt{1-\beta_t}$ - wartość średnia
- β_t - wariancja

Wykorzystując ten wzór można po kolei wyznaczyć obraz oraz szum w każdym koru czasowym t. Jednak każdy kolejny krok musi być wyznaczany kolejno, znając poprzedni. A to oznacza, że aby wyznaczyć obraz oraz szum w kroku czasowym T musimy wykonać T operacji. Jednakże istnieje sposób na wyznaczenie tego w jednym kroku. Można to zrobić odpowiednio przekształcając powyższy wzór.
Na początku musimy zdefiniować zmienną:

$$\alpha_t = 1 - \beta_t \tag{2}$$

oraz skumulowany iloczyn α_t od 0 do t:

$$\bar{a}_t = \prod_{s=1}^{t} a_s \tag{3}$$

Następnie, wykorzystując wzór na reparametryzację rozkładu normalnego:

$$\mathcal{N}(\mu, \sigma^2) = \mu + \sigma \cdot \epsilon \tag{4}$$

gdzie:

- μ - wartość średnia
- σ - odchylenie standardowe
- ϵ - wartość wzięta z rozkładu normalnego o wartości średniej równej 0 oraz odchyleniu standardowym równym 1 ($\epsilon \sim \mathcal{N}(0,1)$)

Przepiszemy równanie (1). Otrzymamy wówczas:

$$q(x_t|x_{t-1}) = \sqrt{1-\beta_t}x_{t-1} + \sqrt{\beta_t}\epsilon \tag{5}$$

A podstawiając wprowadzone wcześniej α_t otrzymamy:

$$q(x_t|x_{t-1}) = \sqrt{\alpha_t}x_{t-1} + \sqrt{1-\alpha_t}\epsilon \tag{6}$$

Możemy jednak rozszerzyć powyższe równanie do poprzednich kroków czasowych i obliczyć obraz x_t bezpośrednio z obrazu x_{t-2} lub x_{t-3} poprzez mnożenie kolejnych alf:

$$q(x_t|x_{t-1}) = \sqrt{\alpha_t\alpha_{t-1}}x_{t-2} + \sqrt{1-\alpha_t\alpha_{t-1}}\epsilon = \sqrt{\alpha_t\alpha_{t-1}\alpha_{t-2}}x_{t-3} + \sqrt{1-\alpha_t\alpha_{t-1}\alpha_{t-2}}\epsilon \tag{7}$$

Podążając tym schematem możemy łatwo zauważyć, że jesteśmy w stanie wyznaczyć obraz x_t bezpośrednio z obrazu początkowego x_0:

$$q(x_t|x_0) = \sqrt{\alpha_t\alpha_{t-1}\dots\alpha_1\alpha_0}x_0 + \sqrt{1-\alpha_t\alpha_{t-1}\dots\alpha_1\alpha_0}\epsilon \tag{8}$$

Podstawiając wprowadzony wcześniej skumulowany iloczyn $\bar{a}_t$ otrzymamy ostateczne równanie na wyznaczenie zaszumionego obrazu x_t znając jedynie x_0:

$$q(x_t|x_0) = \sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon \tag{9}$$

Główny wzór na wyznaczenie zaszumionego obrazu x_t znając jedynie x_0 w procesie propagacji w przód modelu dyfuzyjnego wygląda zatem następująco:

$$q(x_t \mid x_0) = \mathcal{N}(x_t; \sqrt{\bar{\alpha}_t}x_0, (1-\bar{\alpha}_t)I) \tag{10}$$

5.3 Propagacja wsteczna

W propagacji wstecznej model przewiduje obraz x_{t-1} oraz szum dodany w kroku $t-1$ mając obraz x_t. Można to zapisać w taki sposób:

$$p(x_{t-1} \mid x_t) = \mathcal{N}(x_{t-1}; \mu_\theta(x_t, t), \Sigma_\theta(x_t, t)) \tag{11}$$

W takim przypadku mielibyśmy dwie sieci neuronowe parametryzujące rozkład normalny: μ_0 oraz Σ_0. Jednakże, według artykułu [2] można ustawić wartość wariancji jako stałą dla konkretnego harmonogramu β. To powoduje, że nie potrzebujemy przewidywać tej wartości, a więc nie potrzebujemy na nią sieci neuronowej, a wariancja jest zawsze łatwo dostępna.

5.4 Funkcja straty

Funkcją straty w takim modelu może wyglądać następująco:

$$-\log(p_\theta(x_0)) \tag{12}$$

W kontekście modelu dyfuzyjnego odwołujemy się do maksymalizacji prawdopodobieństwa logarytmicznego dla parametrów modelu θ. Celem jest nauczenie modelu generowania próbek x_0 poprzez maksymalizację prawdopodobieństwa (p_0) logarytmicznego, czyli $\log(p_\theta(x_0))$. W celach optymalizacji minimalizuje się jednak negatywne logarytmiczne prawdopodobieństwo, co prowadzi do funkcji straty $-\log(p_\theta(x_0))$. Umożliwia to nauczenie modelu generacyjnego odtwarzania danych wejściowych x_0 z największym możliwym prawdopodobieństwem.

Jednakże w tym przypadku pojawia się problem - $p_\theta(x_0)$ nie jest łatwo obliczalne, ponieważ zależy od wszystkich poprzednich kroków czasowych przed x_0 rozpoczynając od x_T. To oznacza że należy śledzić $T-1$ zmiennych losowych, co jest niemożliwe w praktyce.

Aby to rozwiązać, można obliczyć wariacyjną dolną granicę - pojęcie kluczowe w ramach wariacyjnych metod Bayesowskich, które są stosowane w celu przybliżenia trudnych do obliczenia rozkładów prawdopodobieństwa. Głównym celem jest maksymalizacja dolnej granicy log-prawdopodobieństwa danych. Działa to w sposób następujący: mamy pewną funkcję $f(x)$, której nie potrafimy obliczyć. Jednakże, możemy wyznaczyć funkcję $g(x)$ i udowodnić, że jest ona zawsze mniejsza lub równa $f(x)$: $g(x) \leqslant f(x)$. Wtedy, maksymalizując $g(x) : max(g(x))$ możemy być pewni, że wartość funkcji $f(x)$ również rośnie. Stosując wariacyjną dolną granicę dla wzoru (12) otrzymamy:

$$-\log(p_\theta(x_0)) \leqslant -\log(p_\theta(x_0)) + D_{KL}(q(x_{1:T} \mid x_0) \parallel p_\theta(x_{1:T} \mid x_0)) \tag{13}$$

Zastosowanie wyżej wspomnianego pojęcia dla wzoru (12) działa poprzez odejmowanie dywergencji D_{KL}, która jest miarą tego, jak bardzo dwa rozkłady są do siebie podobne. Jest ona zawsze nieujemna i wygląda następująco:

$$D_{KL}(p \parallel q) = \int_x p(x) \log \frac{p(x)}{q(x)}\, dx \tag{14}$$

A więc zawsze odejmując coś niezerowego od funkcji zawsze otrzymamy coś co jest mniejsza niż orygnalna funkcja. Jednakże mowa tutaj o odejmowaniu dywergencji D_{KL}, a we wzorze (13) występuje ona ze znakiem $+$. Wynika to z faktu, że zamiast maksymalizować, to chcemy minimalizować, a więc do $-\log(p_\theta(x_0))$ znajdującego się po prawej stronie nierówności dodajemy D_{KL} przez co otrzymamy zawsze większą wartość niż sam $-\log(p_\theta(x_0))$ znajdujący się po lewej stronie, przez co możemy minimalizować stratę.

Jednakże po prawej stronie nierówności ciągle pozostaje składnik $-\log(p_\theta(x_0))$ którego nie możemy obliczyć.

Aby go wyeliminować, najpierw rozpiszemy dywergencję D_{KL} jako logarytm stosunku wartości q i $p0$:

$$D_{KL}(q(x_{1:T} \mid x_0) \parallel p_\theta(x_{1:T} \mid x_0)) = \log\left(\frac{q(x_{1:T} \mid x_0)}{p_\theta(x_{1:T} \mid x_0)}\right) \tag{15}$$

Stosując regułę Bayesa:

$$P(A \mid B) = \frac{P(B \mid A)P(A)}{P(B)} \tag{16}$$

Do wyrażenia w mianowniku dla równania (15) otrzymamy:

$$p_\theta(x_{1:T} \mid x_0) = \frac{p_\theta(x_0 \mid x_{1:T})p_\theta(x_{1:T})}{p_\theta(x_0)} \tag{17}$$

Wykorzystując pojęcie wspólnego prawdopodobieństwa:

$$P(A, B) = P(A \mid B)P(B) = P(B \mid A)P(A) \tag{18}$$

dla licznika równania (17) otrzymamy:

$$p_\theta(x_{1:T} \mid x_0) = \frac{p_\theta(x_0 \mid x_{1:T})p_\theta(x_{1:T})}{p_\theta(x_0)} = \frac{p_\theta(x_0, x_{1:T})}{p_\theta(x_0)} = \frac{p_\theta(x_{0:T})}{p_\theta(x_0)} \tag{19}$$

Podstawiając to do równania (15) otrzymamy:

$$\log\left(\frac{q(x_{1:T} \mid x_0)}{\frac{p_\theta(x_{0:T})}{p_\theta(x_0)}}\right) = \log\left(\frac{q(x_{1:T} \mid x_0)}{p_\theta(x_{0:T})}\right) + \log(p_\theta(x_0)) \tag{20}$$

Teraz możemy to podstawić do wzoru (13) i otrzymamy:

$$-\log(p_\theta(x_0)) \leqslant -\log(p_\theta(x_0)) + D_{KL}(q(x_{1:T} \mid x_0) \parallel p_\theta(x_{1:T} \mid x_0)) \tag{21}$$

$$-\log(p_\theta(x_0)) \leqslant -\log(p_\theta(x_0)) + \log\left(\frac{q(x_{1:T} \mid x_0)}{\frac{p_\theta(x_{0:T})}{p_\theta(x_0)}}\right) \tag{22}$$

$$-\log(p_\theta(x_0)) \leqslant -\log(p_\theta(x_0)) + \log\left(\frac{q(x_{1:T} \mid x_0)}{p_\theta(x_{0:T})}\right) + \log(p_\theta(x_0)) \tag{23}$$

$$-\log(p_\theta(x_0)) \leqslant \log\left(\frac{q(x_{1:T} \mid x_0)}{p_\theta(x_{0:T})}\right) \tag{24}$$

W ten sposób pozbyliśmy się czynnika, którego nie byliśmy w stanie obliczyć i otrzymaliśmy wariacyjną dolną granicę, którą możemy minimalizować i znamy wszystkie jej części: licznik jest procesem propagacji w przód rozpoczynającym się na jakimś obrazie z naszego zbioru danych, natomiast mianownik możemy zapisać w następujący sposób:

$$p_\theta(x_{0:T}) = p(x_T) \prod_{t=1}^{T} p_\theta(x_{t-1} \mid x_t) \tag{25}$$

Teraz równanie (25) możemy podstawić do nierówności (24). Wówczas prawa strona nierówności może być zapisana jako:

$$\log\left(\frac{q(x_{1:T} \mid x_0)}{p_\theta(x_{0:T})}\right) = \log\left(\frac{\prod_{t=1}^{T} q(x_t \mid x_{t-1})}{p(x_T) \prod_{t=1}^{T} p_\theta(x_{t-1} \mid x_t)}\right) \tag{26}$$

Dalej przekształcamy powyższe równanie stosując reguły działań na logarytmach:

$$\log\left(\frac{q(x_{1:T} \mid x_0)}{p_\theta(x_{0:T})}\right) = \log\left(\frac{\prod_{t=1}^{T} q(x_t \mid x_{t-1})}{p(x_T) \prod_{t=1}^{T} p_\theta(x_{t-1} \mid x_t)}\right) = -\log(p(x_T)) + \log\left(\frac{\prod_{t=1}^{T} q(x_t \mid x_{t-1})}{\prod_{t=1}^{T} p_\theta(x_{t-1} \mid x_t)}\right) \tag{27}$$

W ten sposób rozbiliśmy logarytm na sumę logarytmów, a drugi składnik tej sumy jest stosunkiem procesu propagacji w przód oraz propagacji wstecznej.

Stosując dalsze przekształcenia możemy zamienić iloczyny liczb logarytmowanych na sumę logarytmów:

$$-\log(p(x_T)) + \log\left(\frac{\prod_{t=1}^{T} q(x_t \mid x_{t-1})}{\prod_{t=1}^{T} p_\theta(x_{t-1} \mid x_t)}\right) = -\log(p(x_T)) + \sum_{t=1}^{T} \log\left(\frac{q(x_t \mid x_{t-1})}{p_\theta(x_{t-1} \mid x_t)}\right) \tag{28}$$

Następnie wyodrębnimy pierwszy składnik sumy otrzymując w ten sposób:

$$= -\log(p(x_T)) + \sum_{t=1}^{T} \log\left(\frac{q(x_t \mid x_{t-1})}{p_\theta(x_{t-1} \mid x_t)}\right) = -\log(p(x_T)) + \sum_{t=2}^{T} \log\left(\frac{q(x_t \mid x_{t-1})}{p_\theta(x_{t-1} \mid x_t)}\right) + \log\left(\frac{q(x_1 \mid x_0)}{p_\theta(x_0 \mid x_1)}\right) \tag{29}$$

Następnie możemy rozpisać licznik sumy wykorzystując stosowaną wcześniej regułę Bayesa:

$$q(x_t \mid x_{t-1}) = \frac{q(x_{t-1} \mid x_t) q(x_t)}{q(x_{t-1})} \tag{30}$$

Jednakże, każdy z elementów tego wyrażenia: $q(x_{t-1} \mid x_t)$, $q(x_t)$ oraz $q(x_{t-1})$ ma wysoką wariancję, ponieważ nie znamy obrazu początkowego x_0. Można jednak znaczocą zmniejszyć tę wariancję warunkując również x_0. Wynika to z tego, że otrzymując jedynie całkowicie zaszumiony obraz nie ma żadnej szansy przewidzieć z jakiego oryginalnego obrazu został stworzony, a więc wariancja będzie wysoka. Jeśli jednak razem z całkowicie

zaszumionym obrazem dostaniemy również obraz oryginalny pula możliwych obrazów z których powstał znacząco spada, a więc wariancja będzie znacznie mniejsza.
Równanie wyglądać będzie zatem następująco:

$$q(x_t \mid x_{t-1}) = \frac{q(x_{t-1} \mid x_t, x_0) q(x_t \mid x_0)}{q(x_{t-1} \mid x_0)} \tag{31}$$

Możemy to podstawić do równania (29). W ten sposób otrzymamy:

$$-\log(p(x_T)) + \sum_{t=2}^{T} \log\left(\frac{q(x_t \mid x_{t-1})}{p_\theta(x_{t-1} \mid x_t)}\right) + \log\left(\frac{q(x_1 \mid x_0)}{p_\theta(x_0 \mid x_1)}\right) =$$

$$= -\log(p(x_T)) + \sum_{t=2}^{T} \log\left(\frac{q(x_{t-1} \mid x_t, x_0) q(x_t \mid x_0)}{p_\theta(x_{t-1} \mid x_t) q(x_{t-1} \mid x_0)}\right) + \log\left(\frac{q(x_1 \mid x_0)}{p_\theta(x_0 \mid x_1)}\right) \tag{32}$$

Następnie rozbijamy sumę na dwie części i otrzymujemy:

$$= -\log(p(x_T)) + \sum_{t=2}^{T} \log\left(\frac{q(x_{t-1} \mid x_t, x_0)}{p_\theta(x_{t-1} \mid x_t)}\right) + \sum_{t=2}^{T} \log\left(\frac{q(x_t \mid x_0)}{q(x_{t-1} \mid x_0)}\right) + \log\left(\frac{q(x_1 \mid x_0)}{p_\theta(x_0 \mid x_1)}\right) \tag{33}$$

Drugi składnik rozbitej sumy może zostać uproszczony do postaci:

$$\sum_{t=2}^{T} \log\left(\frac{q(x_t \mid x_0)}{q(x_{t-1} \mid x_0)}\right) = \log\left(\frac{q(x_T \mid x_0)}{q(x_1 \mid x_0)}\right) \tag{34}$$

Podstawiając to do równania (33) otrzymamy:

$$= -\log(p(x_T)) + \sum_{t=2}^{T} \log\left(\frac{q(x_{t-1} \mid x_t, x_0)}{p_\theta(x_{t-1} \mid x_t)}\right) + \sum_{t=2}^{T} \log\left(\frac{q(x_t \mid x_0)}{q(x_{t-1} \mid x_0)}\right) + \log\left(\frac{q(x_1 \mid x_0)}{p_\theta(x_0 \mid x_1)}\right) =$$

$$= -\log(p(x_T)) + \sum_{t=2}^{T} \log\left(\frac{q(x_{t-1} \mid x_t, x_0)}{p_\theta(x_{t-1} \mid x_t)}\right) + \log\left(\frac{q(x_T \mid x_0)}{q(x_1 \mid x_0)}\right) + \log\left(\frac{q(x_1 \mid x_0)}{p_\theta(x_0 \mid x_1)}\right) \tag{35}$$

Teraz możemy przekształcić dwa ostatnie składniki tej sumy stosując reguły działań na logarytmach - zamieniamy ilorazy liczb logarytmowaych na różnicę logarytmów z tych liczb. Otrzymamy wówczas:

$$\log\left(\frac{q(x_T \mid x_0)}{q(x_1 \mid x_0)}\right) + \log\left(\frac{q(x_1 \mid x_0)}{p_\theta(x_0 \mid x_1)}\right) = \tag{36}$$

$$= \log(q(x_T \mid x_0)) - \log(q(x_1 \mid q(x_0))) + \log(q(x_1 \mid q(x_0))) - \log(p_\theta(x_0 \mid x_1)) = \log(q(x_T \mid x_0)) - \log(p_\theta(x_0 \mid x_1))$$

Podstawiamy to do równania (35) i otrzymujemy analitycznie obliczalne wyrażenie funkcji celu:

$$= -\log(p(x_T)) + \sum_{t=2}^{T} \log\left(\frac{q(x_{t-1} \mid x_t, x_0)}{p_\theta(x_{t-1} \mid x_t)}\right) + \log\left(\frac{q(x_T \mid x_0)}{q(x_1 \mid x_0)}\right) + \log\left(\frac{q(x_1 \mid x_0)}{p_\theta(x_0 \mid x_1)}\right) =$$

$$= \log\left(\frac{q(x_T \mid x_0)}{p(x_T)}\right) + \sum_{t=2}^{T} \log\left(\frac{q(x_{t-1} \mid x_t, x_0)}{p_\theta(x_{t-1} \mid x_t)}\right) - \log(p_\theta(x_0 \mid x_1)) \tag{37}$$

Teraz, dwa pierwsze składniki sumy możemy zapisać wykorzystując używaną wcześniej notację dywergencji D_{KL} i otrzymujemy w ten sposób funkcję celu:

$$= D_{KL}(q(x_T \mid x_0) \parallel p(x_T)) + \sum_{t=2}^{T} D_{KL}(q(x_{t-1} \mid x_t, x_0) \parallel p_\theta(x_{t-1} \mid x_t)) - \log(p_\theta(x_0 \mid x_1)) \tag{38}$$

Jednakże funkcję tę można uprościć. Po pierwsze, pierwszy składnik tej sumy może zostać pominięty, ponieważ q nie ma możliwych do wyuczenia parametrów - jest to proces propagacji w przód, który dodaje szum. $P(x_T)$ natomiast jest losowym szumem gaussowskim. Biorąc też pod uwagę, że q zbiega do rozkładu normalnego możemy być pewni, że cała dywergencja D_{KL} będzie pomijalnie mała.

Funkcja celu wygląda zatem następująco:

$$\sum_{t=2}^{T} D_{KL}(q(x_{t-1} \mid x_t, x_0) \parallel p_\theta(x_{t-1} \mid x_t)) - \log(p_\theta(x_0 \mid x_1)) \tag{39}$$

Jednakże można tę funkcję rozpisać jeszcze bardziej:

$$q(x_{t-1} \mid x_t, x_0) = \mathcal{N}(x_{t-1}; \tilde{\mu}_t(x_t, x_0), \tilde{\beta}_t I) \tag{40}$$

$$p_\theta(x_{t-1} \mid x_t)) = \mathcal{N}(x_{t-1}; \mu_\theta(x_t, t), \Sigma_\theta(x_t, t)) \tag{41}$$

Oraz dalej, rozpisując równanie (40) otrzymamy:

$$\tilde{\mu}_t(\mathbf{X}_t, \mathbf{X}_0) = \frac{\sqrt{\alpha_t}(1 - \bar{\alpha}_{t-1})}{1 - \bar{\alpha}_t}\mathbf{X}_t + \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1 - \bar{\alpha}_t}\mathbf{X}_0 \tag{42}$$

$$\tilde{\beta}_t = \frac{1 - \bar{\alpha}_{t-1}}{1 - \bar{\alpha}_t} \cdot \beta_t \tag{43}$$

Jak wspomniano na początku rozważań matematycznych, β jest stała i ustalona, więc nie bierzemy jej pod uwagę. Skupiamy się zatem jedynie na $\tilde{\mu}_t$.

Równanie (42) może zostać uproszczone przekształcając wzór (9) na wyznaczenie obrazu x_t:

$$x_t = \sqrt{\bar{\alpha}_t}\, x_0 + \sqrt{1 - \bar{\alpha}_t}\, \epsilon \Rightarrow x_0 = \frac{1}{\sqrt{\bar{\alpha}_t}} \left(x_t - \sqrt{1 - \bar{\alpha}_t}\, \epsilon \right) \tag{44}$$

Podstawiając to do równania (42) otrzymamy:

$$\tilde{\mu}_t = \frac{\sqrt{\alpha_t}(1 - \bar{\alpha}_{t-1})}{1 - \bar{\alpha}_t} x_t + \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1 - \bar{\alpha}_t} \frac{1}{\sqrt{\bar{\alpha}_t}} \left(x_t - \sqrt{1 - \bar{\alpha}_t}\, \epsilon \right) = \frac{1}{\sqrt{\alpha_t}} \left(x_t - \frac{\beta_t}{\sqrt{1 - \bar{\alpha}_t}} \epsilon \right) \tag{45}$$

Można zauważyć, że $\tilde{\mu}_t$ nie zależy już od x_0. Co więcej teraz właściwie wszystko co robimy to odejmujemy losowy, przeskalowany szum od obrazu x_t

Teraz możemy zastosować błąd średniokwadratowy pomiędzy właściwym $\tilde{\mu}_t$ (wzór 40) oraz przewidywanym μ_0 (wzór 41):

$$L_t = \frac{1}{2\sigma_t^2} \left\| \tilde{\mu}_t(x_t, x_0) - \mu_\theta(x_t, t) \right\|^2 \tag{46}$$

Podstawiając równanie (45) otrzymamy:

$$L_t = \frac{1}{2\sigma_t^2} \left\| \frac{1}{\sqrt{\alpha_t}} \left(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}} \epsilon \right) - \mu_\theta(x_t, t) \right\|^2 \tag{47}$$

W powyższym równaniu x_t jest wejściem dla modelu. Nie ma potrzeby zatem przewidywać całej wartości wyrażenia $\frac{1}{\sqrt{\alpha_t}} \left(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}} \epsilon \right)$ a jedynie sam szum ϵ.
Następnie μ_0 możemy zapisać jako:

$$\mu_\theta(x_t, t) = \frac{1}{\sqrt{\alpha_t}} \left(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}} \epsilon_\theta(x_t, t) \right) \tag{48}$$

Podstawiając to do wzoru (47) otrzymamy:

$$L_t = \frac{1}{2\sigma_t^2} \left\| \frac{1}{\sqrt{\alpha_t}} \left(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}} \epsilon \right) - \frac{1}{\sqrt{\alpha_t}} \left(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}} \epsilon_\theta(x_t, t) \right) \right\|^2 \tag{49}$$

Upraszczając to wyrażenie i wyłączając wspólny czynnik otrzymamy:

$$L_t = \frac{\beta_t^2}{2\sigma_t^2 \alpha_t (1-\bar{\alpha}_t)} \left\| \epsilon - \epsilon_\theta(x_t, t) \right\|^2 \tag{50}$$

Jednakże, według arykułu [3] pominięcie czynnika $\frac{\beta_t^2}{2\sigma_t^2 \alpha_t (1-\bar{\alpha}_t)}$ powoduje lepszą jakość próbkowania i w ogólności łatwiejszą implementację modelu. Otrzymujemy wzór na funkcję kosztu, która wygląda następująco:

$$L_t = \left\| \epsilon - \epsilon_\theta(x_t, t) \right\|^2 \tag{51}$$

Jednak według artykułu [3] strata nie musi być obliczana tylko dla jednego kroku czasowego t i reprezentować straty dla jednej pojedynczej próbki, ale może być obliczona jako wartość oczekiwana ze wszystkich kroków czasowych, próbek i szumu i może reprezentować uśrednioną stratę dla całego zbioru danych i szumu.
W ten sposób, otrzymujemy ostateczny wzór na funkcję straty, który wygląda następująco:

$$L_{simple} = \mathbb{E}_{t, x_0, \epsilon} \left[\left\| \epsilon - \epsilon_\theta(x_t, t) \right\|^2 \right] \tag{52}$$

Literatura

[1] Schmidhuber, J. (2015). Deep Learning in Neural Networks: An Overview. *arXiv*. https://arxiv.org/pdf/1503.03585

[2] Ho, J., Jain, A., & Abbeel, P. (2020). Denoising Diffusion Probabilistic Models. *arXiv*. https://arxiv.org/pdf/2006.11239

[3] Nichol, A., & Dhariwal, P. (2021). Improved Denoising Diffusion Probabilistic Models. *arXiv*. https://arxiv.org/pdf/2102.09672

[4] Diffusion Models — Paper Explanation — Math Explained. https://www.youtube.com/watch?v=HoKDTa5jHvg

[5] Diffusion models from scratch in PyTorch. https://www.youtube.com/watch?v=HoKDTa5jHvg