ONECLICK AI

FNO

연준모

목차

1. Fourier Embedding
2. Fourier Layer
3. Fourier Neural Operator

1. Fourier Embedding

Fourier Embedding

기존 MLP에선 모델에 고주파를 학습시키기 위해 SIREN을 활용했었다.

$$\phi(x) = \sin(\omega_0 x)$$

값을 증폭시켜 강제적으로 고주파의 형태를 띄게 할 수 있다.

이는 고주파에 둔감한 AI에게 떠먹여 줄 수 있는 강력한 수단이지만, 값이 레이어가 반복될수록 계속 증폭되는 문제점이 있다.

이를 막기위해 다음과 같은 가중치 초기화 식을 활용하여 초기 값을 작게하는 것으로 모델의 발산을 막았다.

$$W \sim \mathcal{U}\left(-\frac{\sqrt{6/n}}{\omega_0}, \frac{\sqrt{6/n}}{\omega_0}\right)$$

1. Fourier Embedding

Fourier Embedding

하지만 Transformer에선?

1. Multi-Head Self-Attention
2. Add & Norm
3. Position-wise Feed-Forward Network (FFN)
4. Add & Norm

이게 인코더 하나에 들어가는 레이어다.

이미 너무 많기에, 아무리 초기가중치를 작게 시도해도 모델은 폭발하게 된다.

그러면 어떻게 해야 트랜스포머 모델에 고주파를 강제로 먹일 수 있을까?

1. Fourier Embedding

Fourier Embedding

정답은 바로 트랜스포머의 위치 인코딩에 있다.

SIREN 은 주기적인 SIN 값을 통해 ReLU나 Tenh 과 같은 다른 활성화 함수들 더 주파수를 학습하기 쉽게 만들어져 있다.

원래 언어 모델이였던 트랜스포머는 sin, cos 위치 인코딩을 통해 서로 겹치지 않는 위치 값들을 얻는다.

이를 응용하여 네트워크의 첫 입력단에서 다음과 같은 식을 통해 주기함수 공간으로 입력 좌표값을 매핑한다.

$$\gamma(\mathbf{v}) = [\cos(2\pi\mathbf{B}\mathbf{v}), \sin(2\pi\mathbf{B}\mathbf{v})]^T$$

그냥 좌표만 들어오면 저주파부터 학습하려는 스펙트럴 편향을 이와같은 방법을 통해 방지한다.

1. Fourier Embedding

Fourier Embedding

$$\gamma(\mathbf{v}) = [\cos(2\pi\mathbf{B}\mathbf{v}), \sin(2\pi\mathbf{B}\mathbf{v})]^T$$

매핑 행렬 $\mathbf{B}$ 를 통해서 SIN, COS의 주파수 값을 조절한다. 보통 50 ~ 100의 큰 값을 넣어 고주파로 신호를 매핑한다.

이는 SIREN의 ω_0와 동일한 역할을 수행한다.

$\mathbf{B}$의 각 원소는 평균이 0이고 분산이 σ^2인 가우시안 정규분포 $\mathcal{N}(0, \sigma^2)$에서 무작위로 추출된다.

σ값을 크게 설정할수록 $\mathbf{B}$행렬 내부에는 절댓값이 큰 숫자인 100, 500 등의 큰 값으로 채워진다.

결과적으로 입력 좌표 $\mathbf{v}$는 $\sin(2\pi \cdot 500 \cdot \mathbf{v})$처럼 극도로 빠르게 진동하는 고주파 파형으로 매핑되는 거다.

1. Fourier Embedding

Fourier Embedding

Neural Tangent Kernel, NTK 관점에서 본 스펙트럴 편향 극복

일반적인 MLP 같은 것들은 NTK 스펙트럴 고유값이 저주파 영역에만 집중되어 있고, 고주파로 갈 수록 기하급수적으로 소실된다.

하지만 입력 데이터에 $\gamma(\mathbf{v})$ 매핑을 적용하면, 신경망의 NTK는 이동 불면성을 가진 커널로 성질이 바뀐다.
이 때 주로 가우시안 커널 형태로 바뀌게 되는 거다.

이때 B의 σ를 키우면 이 커널의 대역폭이 좁아지면서, NTK의 고유값 스펙트럼 전체가 고주파 대역으로 이동한다. 커널의 폭이 바늘처럼 뾰족해 진다는 의미이다.

또 주파수 입장에서도 푸리에 변환의 기본 성질에 의해 공간 도메인에서 좁은 펄스는 주파수 도메인에서 넓은 대역폭을 가지게 되는 거다

따라서 결국 신경망이 가장 민감하게 반응하는 신경망의 공진 주파수 대역이 저주파에서 고주파로 옮겨간다는건데

1. Fourier Embedding

Fourier Embedding

커널이란 -> 데이터 간의 유사도 측정기

$\kappa(x, y)$: 두 데이터 점 x와 y가 얼마나 유사한가 가까운가 를 0과 1 사이의 값으로 측정하는 함수

x와 y가 똑같은 위치면 $\kappa(x, x) = 1$ (유사도 100%)
x와 y가 멀어질수록 $\kappa(x, y) \to 0$ (유사도 0%)

신경망에서 이 유사도는 점 x에서 발생한 오차를 수정할 때, 주변에 있는 점 y의 예측값도 얼마나 같이 끌어당겨서 Smooth하게 바꿀 것인가 즉, 영향력의 반경을 의미하게 된다.

1. Fourier Embedding

Fourier Embedding

가우시안 커널

$$\kappa(x, y) = \exp\left(-\frac{||x - y||^2}{2l^2}\right)$$

l는 Length scale, 대역폭 을 의미하는 파라미터 이다

l이 클 때 (넓은 커널): 두 점 x, y가 꽤 멀리 떨어져 있어도 κ 값이 1에 가깝게 나온다. 신경망은 공간을 아주 부드럽고 뭉툭하게학습합니다. 즉, 저주파만 학습한다.

l이 작을 때 (좁은 커널): 두 점이 아주 조금만 떨어져 있어도 ₩kappa 값이 0으로 꼴아박는다. 자기 자신과 극도로 가까운 점들만 유사하다고 판단한다. 이 덕분에 공간상에서 뾰족하고 급격하게 변하는 값, 즉 고주파를 학습할 수 있게 된다.

1. Fourier Embedding

Fourier Embedding

가우시안 커널 + 푸리에 임배딩

자, 이제 입력 데이터 x에 푸리에 임베딩 $\gamma(x) = [\cos(\mathbf{B}x), \sin(\mathbf{B}x)]$를 통과시켰다고 해보자.

두 점 x와 y를 임베딩한 후, 이 둘의 내적하면

$$\gamma(x)^T\gamma(y) = \cos\big(\mathbf{B}(x-y)\big)$$

이때 $\mathbf{B}$ 행렬의 원소들은 분산이 σ^2인 가우시안 정규분포 $\mathcal{N}(0,\sigma^2)$에서 무작위로 뽑은 값이다.

따라서 위 식의 기댓값을 구하게 되는데, 통계학에서 정규분포의 특성 함수 공식에 의해
다음 식이 성립하게 된다.

$$E_{\mathbf{B}\sim\mathbb{N}(\mathbb{0},\sigma^2)}\big[\cos\big(\mathbf{B}(x-y)\big)\big] = \exp\left(-\frac{1}{2}\sigma^2\big||x-y|\big|^2\right)$$

1. Fourier Embedding

Fourier Embedding

그래서 커널이 좁아진다는 것은

일반 가우시안 커널: $\exp\left(-\frac{||x-y||^2}{2l^2}\right)$

푸리에 임베딩 커널: $\exp\left(-\frac{1}{2}\sigma^2||x-y||^2\right)$

수학적으로 $\sigma^2 = \frac{1}{l^2}$이라는 정확한 역수 관계가 성립하므로 $\sigma = 1/l$ 이다.

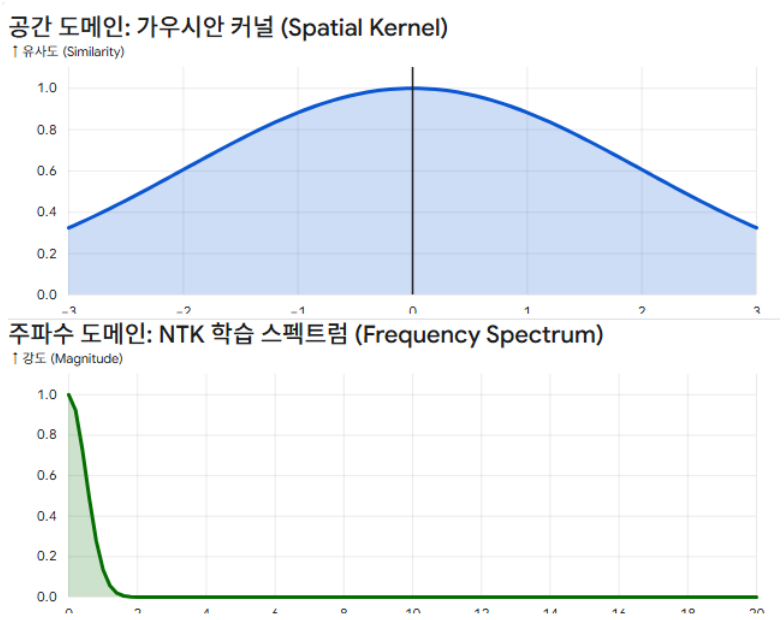
우리가 푸리에 임베딩에서 고주파를 쏘아주기 위해 **σ를 크게 키우면**, 가우시안 커널의 관점에서는 길이 척도인 **l이 엄청나게 작아진다.**

l이 작아진다는 것은, 커널 함수의 모양이 넓은 종 모양에서 **바늘처럼 뾰족하고 좁은 형태**로 변한다는 의미이다.

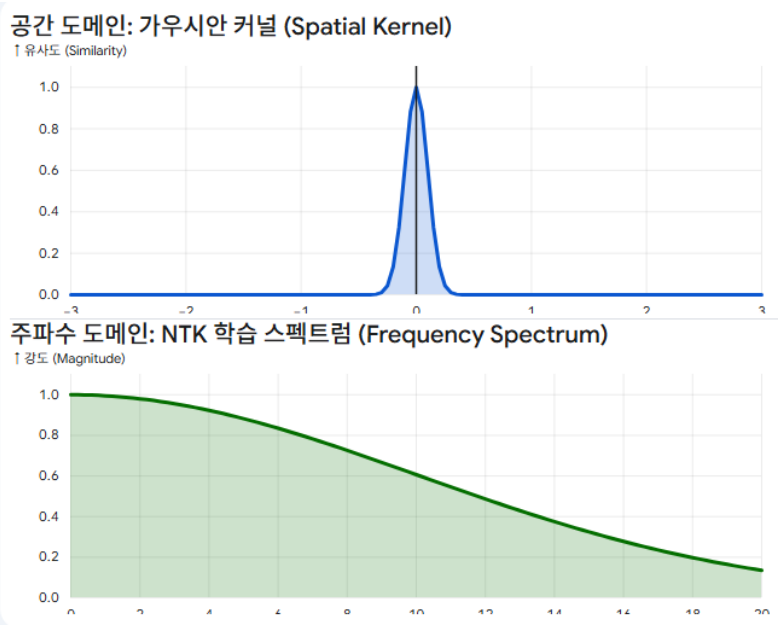
1. Fourier Embedding

Fourier Embedding

공간 도메인에서 커널이 좁아지면(바늘처럼 변하면), 푸리에 변환의 성질에 의해 주파수 도메인에서는 스펙트럼이 무한히 넓게 퍼진다. 이게 NTK 스펙트럼이동 되시겠다.



시그마 0.5



시그마 10

1. Fourier Embedding

Fourier Embedding

Neural Tangent Kernel, NTK 관점에서 본 스펙트럴 편향 극복

쉽게 말하면 저주파 좋아하면 뭐 하냐 입력 데이터에서 제일 낮을 주파수가 100GHz 이래버리면

모델도 뭐라도 먹고 살아야 하니까 학습하는 주파수가 저주파에서 고주파로 옮겨진다 그런 내용이다.

원래는 저주파만 잘 먹었었는데 입력 데이터가 강제로 다 고주파가 되어 버리니까
고주파라도 퍼먹는다고 생각하면 쉽다.

1. Fourier Embedding

Fourier Embedding

혀튼 SIREN을 사용했을 때 처럼 데이터가 주기적이게 된다. 물론 여기선 입력 데이터 이긴 하지만.

또한 SIREN은 매 레이어를 통과할 때 마다 값이 증폭되었지만, 푸리에는 그렇지 않기에 학습에 있어서 더 안정적이다.

2. Fourier Layer

Fourier Layer

앞서 우리는 푸리에 임배딩으로 입력값을 주기적이게 만들었다.

이어서 푸리에 레이어는 신경망의 연산 과정 자체를 공간 도메인이 아니라 주파수 도메인에서 진행하게 된다.

이것도 ConV 응용인데,

들어가기에 앞서 우리는 Global Integral Operator $\mathcal{K}$에 대해 알고 가야 한다

2. Fourier Layer

Global Integral Operator

신경망의 PDE 해를 위한 점을 찍으려면, 공간 내에서 모든 점들의 상호작용을 계산하는 전역적 적분 연산자가 필요하다.

$$\mathcal{K}(v)(x) = \int_D \kappa(x,y)v(y)dy$$

$v(y)$: 위치 y에서의 입력 특징 맵

현재 신경망 레이어에서, 공간상의 특정 좌표 y에 존재하는 물리량.
위치 y에서의 전기장 벡터값이나 유체역학이면 y에서의 속도나 압력값이다.

$\kappa(x,y)$: 커널 함수

위치 y에 있는 정보가, 목표 위치 x의 상태를 결정하는 데 얼마나 큰 영향을 미치는가?
를 결정하는 영향력 지표

2. Fourier Layer

Global Integral Operator

$$\mathcal{K}(v)(x) = \int_D \kappa(x,y)v(y)dy$$

적분

관심 있는 전체 물리적 공간에 존재하는 무수히 많은 모든 점 y들에 대하여,
위의 $\kappa(x,y)v(y)$값을 싹 다 더하라는 뜻

CNN은 3x3 필터를 써서 x 주변의 아주 좁은 이웃 정보만 더하지만, 여기서는 전체 도메인을 모두 Sweep하며 더하기 때문에, 단 한 번의 레이어 연산만으로 공간 전체의 상호작용을 파악할 수 있다.

$\mathcal{K}(v)(x)$ $\mathcal{K}(v)$는 적분 연산자. 새로운 함수 출력장 만들었음을 의미.

(x): 그렇게 만들어진 새로운 함수를, 우리가 관심 있는 특정 타겟 위치 x에서 평가한 최종 값

2. Fourier Layer

Global Integral Operator

$$\mathcal{K}(v)(x) = \int_D \kappa(x,y)v(y)dy$$

공간의 왼쪽 끝에서는 파동이 빨리 퍼지고 오른쪽 끝에서는 느리게 퍼진다면
위치 x, y가 어디냐에 따라 매번 다른 규칙 κ를 적용해야 한다.

하지만 헬름홀츠나 나비에-스토크스 같은 애들은 균일한 매질 안에서 공간의 어디서나 동일하게 작동한다.

때문에 위치백터 말곤 무시해도 된다.
잔잔한 호숫물이 서울에 있던 부산에있던 돌맹이 던졌을 때 퍼져나가는
파동의 모양이 같다는 것과 같은 의미 이다.

따라서 커널을 $\kappa(x,y)$에서 거리/변위 벡터에만 의존하는 $\kappa(x-y)$로 축소하여 가정해도 상관 없다.

2. Fourier Layer

Global Integral Operator

$$\mathcal{K}(v)(x) = \int_D \kappa(x,y)v(y)dy \quad \text{->} \quad \mathcal{K}(v)(x) = \int_D \kappa(x-y)v(y)dy$$

어? 근데 수식이 conV 수식이랑 닮았네?

$$(f * g)(x) = \int_{-\infty}^{\infty} f(x-\tau)g(\tau)d\tau$$

이건 CNN 편 참고해라

$f \rightarrow \kappa$ (커널 함수)
$g \rightarrow v$ (입력 특징 함수)
$\tau \rightarrow y$ (적분 변수, 공간상의 모든 점)

-> $(\kappa * v)(x) = \int_D \kappa(x-y)v(y)dy$

즉, 거리만 본다고 치면 합성곱으로 날먹할 수 있다는 것이다.

2. Fourier Layer

Global Integral Operator

자 원래 적분식을 보자

$$(\kappa * v)(x) = \int_D \kappa(x-y)v(y)dy \qquad \text{->} \qquad \mathcal{F}(\kappa * v) = \mathcal{F}(\kappa) \cdot \mathcal{F}(v)$$

공간에서 무식하게 N^2번 적분하는 대신

입력을 FFT 주파수 도메인으로 보내고 $\mathcal{O}(N \log N)$ <- 이거 지수 역함수 로그 아님 복잡도의 log는 반씩 줄어든다 그런 의미임

주파수 공간에서 학습 가능한 파라미터 $\mathcal{F}(\kappa)$와 단순히 원소별 곱셈을 한 뒤 $\mathcal{O}(N)$

다시 iFFT로 복원하면 $\mathcal{O}(N \log N)$

이렇게 연산량 쌀먹하겠다는 강한 의지인 것이다.

2. Fourier Layer

Fourier Layer

1. FFT Fast Fourier Transform 로 입력 특징맵을 주파수 도메인으로 변환한다. $\mathcal{F}(x)$

입력 필드 v(x) 특정 채널 c를 가진 1D/2D/3D 그리드 데이터 를 공간 도메인에서 주파수 파수 도메인으로 변환한다.

컴퓨터는 이산화된 그리드 데이터를 다루므로 이산 푸리에 변환 DFT 을 수행하며, 알고리즘적으로는 FFT를 사용한다.

$$\widehat{v_k} = \sum_{j=0}^{N-1} v_j e^{-i\frac{2\pi}{N}kj} \quad (k = 0,1,\dots,N-1)$$

N: 1차원 그리드 크기
공간의 좌표 x_j에서의 값 v_j

2. Fourier Layer

Fourier Layer

1. FFT Fast Fourier Transform 로 입력 특징맵을 주파수 도메인으로 변환한다. $\mathcal{F}(x)$

$$\widehat{v_k} = \sum_{j=0}^{N-1} v_j e^{-i\frac{2\pi}{N}kj} \quad (k = 0,1,\dots,N-1)$$

변환된 텐서 $\hat{v} \in C^{N\times c}$는 복소수 공간에 존재하게 된다.

k는 주파수 모드를 의미하며, k가 작을수록 파장이 긴 거시적 형태 저주파 를,
클수록 미세한 변화 고주파 를 나타낸다.

2. Fourier Layer

Fourier Layer

2. 선형 변환한다.

주파수 도메인에서 학습 가능란 가중치 텐서 $\mathbf{R}$ 과 곱해진다.
여기서 불필요한 고주파 노이즈를 필터링 하거나, 특정 주파수를 증폭시킬 수 있다.

1. 주파수 절단

물리적 파동이나 유체 현상은 에너지의 대부분이 저주파/중주파에 집중되어 있다.

따라서 전체 주파수 N개를 모두 쓰지 않고, 사용자가 지정한 하이퍼파라미터인 최대 주파수 모드 k_{max} 까지만 남기고 나머지 고주파는 과감히 잘라낸다.

$$\widehat{v'} = \text{Truncate}(\hat{v}, k_{max}) \in C^{k_{max} \times c}$$

2. Fourier Layer

Fourier Layer

2. 선형 변환한다.

2. 복소수 가중치 행렬곱

잘라낸 주파수 성분 $₩hat{v}'$에 학습 가능한 복소수 가중치 텐서 $\mathbf{R} \in C^{k_{max} \times c \times c_{out}}$을 곱한다.

각 주파수 모드 k에 대하여, 채널을 믹싱하는 행렬곱이 수행된다

$$\widehat{u_k} = \sum_{c'=1}^{c} \mathbf{R}_{\mathbf{k},\mathbf{c}',\cdot} \widehat{v'_{k,c'}}$$

이 $\mathbf{R}$ 텐서가 바로 PDE의 지배 방정식을 주파수 공간에서 학습한 매개변수 이다.

해상도 N에 독립적(k_{max}까지만 연산하므로)이기 때문에 Zero shot 이 가능한 이유가 바로 이 식에 있는거다.

2. Fourier Layer

Fourier Layer

3. iFFT 다시 공간 도메인으로 돌려버린다. $\mathcal{F}^{-1}(\cdot)$

앞서 믹싱이 끝난 복소수 텐서 $\hat{u}$를 다시 원래의 공간 도메인으로 복원한다.

$$u_j = \frac{1}{N}\sum_{k=0}^{N-1} \widehat{u_k}\, e^{i\frac{2\pi}{N}kj}$$

물리적 공간의 해는 항상 실수여야 하므로, 변환 결과의 실수부만 취한다

$$u(x) = \mathrm{Re}\left(\mathcal{F}^{-1}(\hat{u})\right)$$

2. Fourier Layer

Fourier Layer

하지만, 위 적분 연산 $\mathcal{K}$ 만으론 k_{max}로 고주파를 싹둑 잘라냈기 때문에
BC의 날카로운 변화같은 디테일이 뭉개지게 된다.

고주파를 전부 날려버렸기 때문이다. 대신 저주파의 연산량은 적게, 전체 공간을 한번에 훑으며
균등하게 파악한다

하지만 이러면 안된다 왜냐 고주파의 디테일이 중요하기 때문.
이를 위해 현재 레이어에 입력된 값 l을 이 레이어의 출력에 더해주는 것이다.

비슷한게 있죠 VIT에

2. Fourier Layer

Fourier Layer

결과적으로, 푸리에 레이어가 통과되면 다음과 같이 나오게 된다.

주파수 도메인에서 학습 가능란 가중치 텐서 $\mathbf{R}$ 과 곱해진다.

$$z_{l+1} = \sigma\left(\mathbf{W}z_l + \mathcal{F}^{-1}\left(\mathbf{R}\cdot\mathcal{F}(z_l)\right)\right)$$

$v^{(l)}$: 입력 특징 맵

W : 이 레이어에 들어오는 입력값

$\mathcal{F}^{-1}\left(\mathbf{R}\cdot\mathcal{F}(\cdot)\right)$: 글로벌 물리 법칙을 파악하는 주파수 도메인 적분 연산자

σ: 비선형 활성화 함수 주로 GELU를 사용

3. Fourier Neural Operator

Fourier Neural Operator

이제, 앞서 만든 푸리에 레이어만 엄청나게 쌓아 올려보자. 그러면 FNO다.

많이 했던 PINN이 특정 조건에 대한 딱 그 정답 하나만 찾는데 급급했다면,
FNO는 공간 자체를 학습하게 된다. 때문에 어떤 입력 형상이 들어오던, 추가
적인 수렴 없이 바로 결과를 볼 수 있다.

또한 저해상도에서 학습시킨 모델을 고해상도 메시에 그대로 적용해도 상관이 없다 여긴 점의 세상이니까

3. Fourier Neural Operator

Fourier Neural Operator

1. 입력: 간 좌표 (x, y)나 초기 조건 등의 물리적 입력이다.

2. 푸리에 임베딩: 입력 데이터에 대해 푸리에 임배딩을 실행한다

3. Lifting Layer: 얕은 MLP로 입력 타원을 고차원인 64, 256까지 올린다.
Transformer 의 패치 임배딩과 같다.

4. Fourier Layers: FFT ~ 1x1 Conv 의 푸리에 레이어를 4 ~ 5번 정도 반복한다.

5. Projection Layer: 또 얕은 MLP 사용해서 차원 다시 낮춰준다. 원하는 출력값과 같은 차원으로 맞추면 된다.

6. 출력: Loss 계산, 역전파 이렇게 마무리 된다.

감사합니다