Visual Fourier Prompt Tuning

> 💡 **논문 세줄 요약**
>
> 1. **VFPT**(Visual Fourier Propmt Tuning)는 대규모 Transformer Vision 모델을 효율적으로 미세 조정하기 위해 제안된 **Parameter Efficient**한 방법
> 2. **FFT**(Fast Fourier Transform)을 프롬프트 임베딩에 통합하여 공간 및 주파수 정보를 함께 활용 → **Pre-training - Fine-tuning 간 데이터 불일치 문제 해결**
> 3. **VFPT**는 **0.57%의 적은 파라미터**로도 VTAB-1k에서 평균 73.20%의 정확도 달성

- 논문 링크 : [link]
- 깃허브 링크 : [link]

1. Introduction

- **Prompt Tuning 기법의 확장**
  - Language Prompt Tuning(NLP) → **Vision Prompt Tuning(Vision)**
  - Classification, Segmentation, Detection 등 다양한 영역에서 뛰어난 성능
  - **But, Pre-Training 과 Fine-Tuning 학습 데이터 간 격차가 클수록 성능 저하**

> 💡 **Prompt Tuning은 서로 다른 데이터셋에서도 일반화가 가능할까 ??**

연구 동기 :

인간의 시각 인지는 다양한 도메인 정보를 빠르게 통합 & 대응 → 사람의 시각 인지 능력을 모방

- **Fourier Transform**에서의 영감
  - 사람의 시각 인지 방식과 Visual Prompt Tuning은 **다양한 도메인 정보(시간, 주파수)등을 통합**한다는 점에서 유사
  - **FFT**(Fast Fourier Transform)
    - **Signal → Frequency** 영역으로 변환하여 정보를 추출
  - 즉, **기존의 Prompt에 주파수 영역의 도메인 정보와 통합**함으로써 다양한 데이터 간 불일치에도 Robustness를 기대할 수 있음

> 💡 **Fourier Transform을 Prompt Tuning에 통합하여 인간 시각 메커니즘을 모방할 수 있을까?**

- **VFPT**(Visual Fourier Prompt Tuning)
  - **주파수**(Frequency) 정보를 **Prompt Embedding에 통합**
    - **공간(Spatial) + 주파수(Frequency) → 동시에 활용**
  - 새로운 Prompt Tuning 전략 제시

- **VFPT의 장점 3가지**
  1. **Simplicity** : FFT를 이용한 Prompt Tuning은 직관적이고 구현인 간단
  2. **Generality** : 주파수 정보를 도입함으로써, Prompt의 Embedding Space가 확장 → 다양한 데이터셋과 작업에서 향상된 성능
  3. **Interpretability** : 주파수 정보가 도입된 Prompt는 Transformer 계열의 입력 공간에서 높은 집중도를 보임 → 시각적으로 해석이 용이

2. Related Work

2.1 Visual Parameter-efficient Finetuning

- Vision 모델의 규모가 급격히 증가, 특히 ViT 등장 이후 **pretrain-then-finetune** 패러다임에서 **PEFT**(Parameter-Efficient Fine-Tuning) 방법 개발이 중요해짐
- **PEFT**
  1. **Partial Tuning** → 일부 Layer만 업데이트
  2. **Extra Module** → LoRA 등
  3. **Prompt Tuning** → **입력단**에 Learnable Prompt를 삽입

> ### ❎ Partial Tuning & Extra Module의 한계
>
> 1. **Unsatisfactory Performance** (성능 부족)
>   - **Full Fine-tuning** 대비 아쉬운 성능
> 2. **Model-oriented Design** (아키텍처 의존적)
>   - 대부분은 특정 백본 구조에 종속 → **다양한 백본에 적용이 어려움**

💡 **Prompt Tuning은 간단하면서도 일반적, Vision 분야에서 새로운 패러다임으로 자리 잡고 있음**

→ **입력 Sequence에만 Learnable Parameter 추가** (전체 모델 재학습 X)

현재의 Prompt Tuning 관련 연구들은 **Engineering Optimizations**에 집중

- 파라미터 수 감소
- 다양한 task에서의 범용성 향상

그러나, 성능 향상을 위해 **복잡한 제약 조건** 혹은 **기능 추가** 등을 도입 → Prompt Tuning의 초기 취지와 ❌

본 논문은 Human Visual Intelligence의 관점에서 Prompt Tuning을 탐색하고, **Prompt Tuning의 간결성을 유지**하는 데 중점을 둠.

또한, Visual Prompt Tuning과 Visual Instruction Tuning은 서로 뚜렷이 구별된다고 강조.

- **Instruction Tuning** : 모델의 지시 수행 능력 향상을 목표로 함
  - 명령을 이해하고 수행할 수 있도록 모델을 만드는 과정
- **Prompt Tuning** : 입력 파라미터를 조정하여 성능을 향상
  - 입력단에 학습 가능한 파라미터를 추가하여 성능 향상에 집중

2.2 Fast Fourier Transform in Vision

- FFT란 무엇인가 ?
  - FFT는 이산 푸리에 변환(Discrete Fourier Transform) 및 그 역변환을 계산하는 수학적 알고리즘으로, **다양한 신호(이미지, 레이더)의 주파수 분석에 핵심적인 역할**
  - Vision에서 복잡한 Spatial 데이터를 Frequency 도메인으로 변환 → **노이즈나 고차원 데이터에서 중요한 특징을 추출**하는 데 유용
  - **즉, 이미지에서 중요한 feature를 뽑아내어 도메인 일반화 성능 향상에 기여** (기존에는 전처리 과정 등에 주로 활용)

FFT와 Visual Prompt Tuning의 결합은 미개척 분야

따라서, 본 논문은 제한된 기존 연구의 범위를 넓히고, **FFT와 Prompt Tuning을 완전히 통합하는 방향을 제시**

- 다양한 데이터셋에 대한 **Adoption 능력 향상**
- Transformer 내부에서 **강한 상관성 확인**

---

3. Methodology

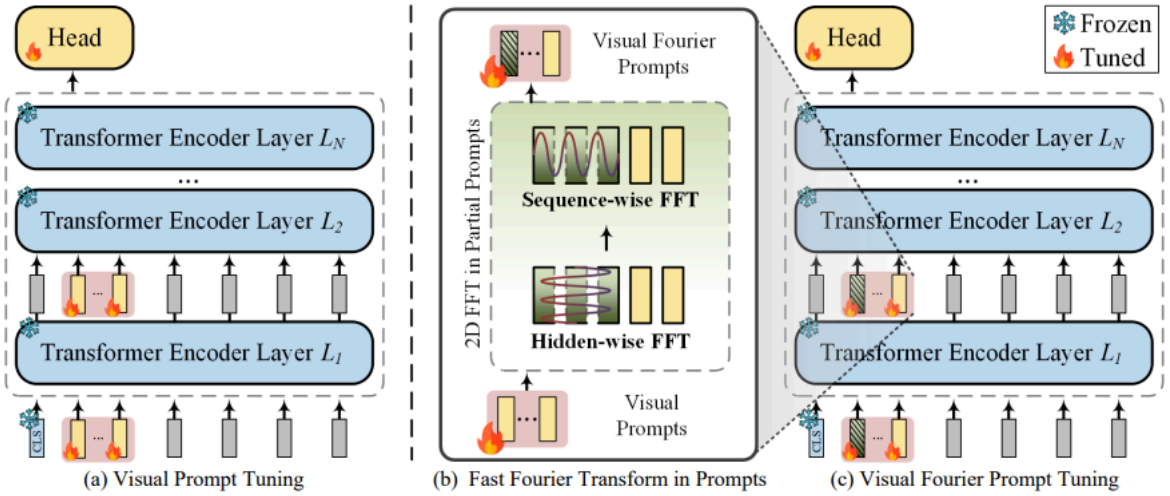


Figure 1: **Overview of VPT** *vs.* **VFPT (ours) frameworks.** (a) Original Visual Prompt Tuning. (b) 2D Fast Fourier Transform operations in partial visual prompts along hidden and sequence length dimensions. (c) The overall architecture of our proposed VFPT (see §3.2).

VPT vs VFPT

3.1. Preliminary

🔷 Visual Prompt Tuning

- 사전학습 된 Transformer 모델 T (N layers)가 있을 때, **VPT**(Visual Prompt Tuning)의 목표는 입력 Sequence에 몇 개의 **Learnable Embedding Vector**만 추가하여 **모델 $\hat{T}$ 로 Fine-tuning**
- $P = \{P^1, P^2, \dots, P^N\}$: **Learnable Prompt** (P^i는 i_{th} 번째 layer의 learnable visual prompt)

> **위 내용에 기반하여 Encoder Layer들은 다음과 같이 정의**

$$Z^1 = L_1(P^1, E)$$

$$Z^i = L_i(P^i, Z^{i-1}), i = 2, 3, ..., N$$

- 임베딩 된 이미지 패치 E는 **사전학습 된 Embedding Layer로 추출**
- Z^i는 i_{th} layer를 통과한 Contextual embeddings

- L_i, E : frozen Parameters
- P^i : trainable Parameters (small proportion of the total parameters)

🔷 Visual Fourier Prompt Tuning

- **FFT**는 **Discrete Fourier Transform**을 계산하는 알고리즘, **일정 간격으로 샘플링 된 신호를 주파수 도메인으로 변환**

$$\mathcal{F}(x) = X_k = \sum_{n=0}^{N-1} x_n \, e^{-i2\pi \frac{k}{N} n}, \quad 0 \leq k \leq N-1$$

- x_n : 시간 영역 입력 신호 (동등한 간격의 finite Sequence)
- X_k : 주파수 영역 출력 값 (x_n과 동등한 간격의 Samples)
- DFT의 계산 복잡도는 $O(n^2)$이나 **Cooley-Tukey FFT** 알고리즘으로 $O(nlogn)$

3.2. Visual Fourier Prompt Tuning

- Prompt Tuning은 **파라미터 효율적**이나, 사전학습과 미세조정 **데이터 간 차이가 클 경우 성능이 저하**.
- 기존 Prompt Tuning은 대부분 **Spatial 정보에 집중 → 새로운 task에 적응이 어려움**

🔷 VFPT

- **Spatial** 정보 뿐만 아니라 **Frequency** 정보를 반영하기 위해, Prompt Vector 일부에 대해 **2D FFT**를 적용
- **Spatial + Frequency 정보를 통합**하여 특징 분리 능력을 강화

1. Encoder Layer i_{th}의 프롬프트 $P^i = \{p_1^i, p_2^i, \ldots, p_M^i\}$
2. m개를 선택해 **Fourier Prompt**로 변환 (나머지는 원래의 Prompt로 유지)
   - 변환 비율 : $\alpha = \frac{m}{M}, \quad (0 \leq \alpha \leq 1)$

$$P_{\mathcal{F}}^i = \Re\left(\mathcal{F}_{\text{seq}}\left(\mathcal{F}_h\left(\left[p_1^i, p_2^i, \ldots, p_m^i\right]\right)\right)\right)$$

- $\mathcal{F}_h$, $\mathcal{F}_{seq}$: 각각 Hidden Dimensions, Sequence에 대한 **1D FFT**
  - $\mathcal{F}_{seq}(\mathcal{F}_h(x))$와 $\mathcal{F}_h(\mathcal{F}_h(seq))$는 **수학적으로 동일** (1차원에서의 FFT의 특징 때문)
- $\Re(\cdot)$: 실수부만 취하기 위함 (복소수는 Self-Attention 연산에 사용할 수 없음)

결합된 최종 프롬프트

- 사전학습 구조를 유지하기 위해서 **Only Prompt Embeddings만 변화**

$$\hat{P}^i = [P^i_{\mathcal{F}}, p^i_{m+1}, ..., p^i_M]$$

```
# Visual Prompts
x = torch.cat((    x[:, :1, :],
            prompt_dropout(prompt_proj(prompt_embeddings).expand(B, -1,
-1)),
            x[:, 1:, :]), dim=1)

# Visual Fourier Prompts
x = torch.cat((    x[:, :1, :],
            torch.fft.fft(torch.fft.fft(
            prompt_dropout(prompt_proj(prompt_embeddings).expand(B, -1,
-1)),
                                    dim=-1),dim=-2).real,
            x[:, 1:, :]), dim=1)
```

개인적인 의문점

- 현재는 α 값에 따라 Spatial 정보와 Frequency 정보를 적절한 비율로 섞어 사용
- **Spatial과 Frequency 정보를 Element-Wise 혹은 Concatenation 하여 사용할 순 없을까 ??**

🔷 VFPT 핵심 특성

- **Simplicity**

    - VPT의 단순한 구조 유지 + **코드 몇 줄로 구현 가능**
    - FFT의 **낮은 계산 복잡도**($O(nlogn)$)
    - 기존 기법들은 **복잡한 아키텍처 변경** or **Self-Attention 추가**
- **Generality**
    - Spatial과 Frequency는 **상호보완적**
        - **Spatial** : 디테일한 물체 경계나 구조 인식
        - **Frequency** : 노이즈 분리, 조명 변화에 강건
    - *“leading to a more comprehensive feature understanding”*
- **Interpretability**
    - VFPT에서 주파수 변환된 Prompt가 시각적인 패턴과 일관된 형태로 나타나며
    - Attention Map과의 일치도 향상
    - Visual Prompt의 역할과 효과를 **직관적으로 시각화 및 이해 가능**

---

4. Experiment

4.1. Experiment Setup

Datasets

2가지 Image Classification 벤치마크에서 실험 진행

- **VTAB-1k** (19개의 Visual Task Adaptaion 과제를 포함)
    1. **Natural** : 일반 카메라 촬영 이미지
    2. **Specialized** : 특수 장비로 촬영된 이미지
    3. **Structured** : 거리 측정 등 기하적인 이해가 필요한 작업 (Natural과 Specialized에 비해 사전학습 데이터셋과의 차이가 크다고 함)
- **FGVC** (Fine-Grained Visual Classification)
    - CUB-200-2011, NABirds, Oxford Flowers, Stanford Dogs, Stanford Cars 로 이루어진 미세 분류 데이터셋

Baselines

공정한 비교를 위해 다양한 Parameter Efficient Fine-Tuning 기법들과 비교

- 사용 BackBone
  - ViT
  - Swin Transformer
- MAE, MoCo v3라는 자기지도학습 기반 모델에서도 사용

4.2. Main Results

논문의 저자들은 **VFPT**의 효과를 **두 가지 관점에서 입증**

🔶 Superior Performance

- VFPT는 다양한 데이터셋(특히, 데이터 간 격차가 큰 경우에서) **유의미한 성능 향상** → **우수한 일반화 성능** 입증

🔶 Fourier Contribution

- 실험 결과, 데이터 간 격차(Disparity)가 클수록 VFPT에서 **Fourier Component의 비율을 높이는 것이 유리** → Fourier 정보가 성능 향상에 중요한 역할을 함

🔷 Definition of disparity

다른 문헌에 기반하여, **FID(Fréchet Inception Distance)**를 사용하여 사전학습 데이터셋과 미세조정 데이터셋 간 **Disparity(분포차이)**를 측정

- Natural ↔ ImageNet : 유사도 높음 → FID 낮음
- Structured ↔ ImageNet : 유사도 낮음 → FID 높음

🔷 VFPT on ViT

Table 1: **Image classification accuracy for ViT-Base/16 [23]** pretrained on supervised ImageNet-21k. Following [4, 5], we report the average test accuracy (three runs) on FGVC [4] and VTAB-1k [78] benchmarks, and "Number of Wins" in [·] compared to full fine-tuning (Full) [92]. ► denotes the method with highest "Number of Wins" compared to Full. We further report "Number of Wins to VPT" in {·}. "Tuned/Total" is the average percentage of tuned parameters required by 24 tasks. "Scope" indicates the tuning scope of each method. "Additional parameters" is the existence of parameters in addition to the pretrained backbone and linear head. **Bold** and **Underline** indicate the best and the second best results. VFPT outperforms full fine-tuning in **22 of 24** instances with fewer trainable parameters and beats VPT in **23 of 24** cases with lower parameters. † denotes methods using soft filtered prompts to reduce the parameter usage in learnable visual prompts, requiring specialized devices to facilitate acceleration. Per-task results are available in Appendix. Same for Table 2 and 3.

ViT-Base/16 [23] (85.8M)	Tuned/ Total	Scope Input	Scope Backbone	Extra params	FGVC [4] [5]	VTAB-1k [78] [19] *Natural* [7]	*Specialized* [4]	*Structured* [8]	Mean Total
Full [CVPR22][92]	100.00%		✓		88.54%	75.88%	83.36%	47.64%	65.57%
Linear [CVPR22][92]	0.08%				79.32% [0]	68.93% [1]	77.16% [1]	26.84% [0]	52.94%
Partial-1 [NeurIPS14][93]	8.34%				82.63% [0]	69.44% [2]	78.53% [0]	34.17% [0]	56.52%
MLP-3 [CVPR20][94]	1.44%			✓	79.80% [0]	67.80% [2]	72.83% [0]	30.62% [0]	53.21%
Sidetune [ECCV20][31]	10.08%		✓	✓	78.35% [0]	58.21% [0]	68.12% [0]	23.41% [0]	45.65%
Bias [NeurIPS17][30]	0.80%		✓		88.41% [3]	73.30% [3]	78.25% [0]	44.09% [2]	62.05%
Adapter [NeurIPS20][32]	1.02%		✓	✓	85.46% [1]	70.67% [4]	77.80% [0]	33.09% [0]	62.41%
LoRA [ICLR22][35]	—		✓	✓	89.46% [3]	78.26% [5]	83.78% [2]	56.20% [7]	72.25%
AdaptFormer [NeurIPS22][95]	—		✓	✓	—	80.56% [6]	84.88% [4]	58.83% [7]	72.32%
ARC_{att} [NeurIPS23][96]	—		✓	✓	89.12% [4]	80.41% [7]	**85.55%** [3]	58.38% [8]	72.32%
VPT-S [ECCV22][4]	0.16%	✓		✓	84.62% [1]	76.81% [4]	79.66% [0]	46.98% [4]	64.85%
VPT-D [ECCV22][4]	0.73%	✓		✓	89.11% [4]	78.48% [6]	82.43% [2]	54.98% [8]	69.43%
EXPRES [CVPR23][97]	—	✓		✓	—	79.69% [6]	84.03% [3]	54.99% [8]	70.20%
† E2VPT [ICCV23][5]	0.39%	✓	✓	✓	89.22% [4]	80.01% [6]	84.43% [3]	57.39% [8]	71.42%
► Ours	0.66%	✓		✓	**89.24%** [4] {4}	**81.35%** [6] {7}	84.93% [4] {4}	**60.19%** [8] {8}	**73.20%**

- 백본 모델 : **ViT-B/16**
- 사전학습 : **ImageNet-21k**
- test Dataset : **VTAB-1k, FGVC**

주목할 점

- **VTAB-1k**에서 **Full Fine-tuning**보다 평균 정확도가 +7.63% 더 높음
- **VPT-S** 대비 평균 정확도가 +6.71% 높으면서, 0.5%의 파라미터만 추가 계산
- **Fine-Grained**에서도 최고 성능이며, VPT-S 대비 +4.62%

Table 2: **Image classification accuracy for Swin-Base [24]** pretrained on supervised ImageNet-21k.

Swin-Base [24] (86.7M)	Tuned/ Total	VTAB-1k [78] [19] *Natural* [7]	*Specialized* [4]	*Structured* [8]
Full [ICLR23][98]	100.00%	79.10%	86.21%	59.65%
Linear [ICLR23][98]	0.06%	73.52% [5]	80.77% [0]	33.52% [0]
Partial-1 [NeurIPS14][93]	14.58%	73.11% [4]	81.70% [0]	34.96% [0]
MLP-3 [CVPR20][94]	2.42%	73.56% [5]	75.21% [0]	35.69% [0]
Bias [NeurIPS17][30]	0.29%	74.19% [2]	80.14% [0]	42.42% [0]
VPT [ECCV22][4]	0.25%	76.78% [6]	83.33% [0]	51.85% [0]
† E2VPT [ICCV23][5]	0.21%	83.31% [6]	84.95% [2]	57.35% [3]
► Ours	0.27%	**84.53%** [7] {5}	**86.15%** [2] {4}	**58.21%** [3] {6}

- 백본 모델 : **Swin-Base**
  - Swin의 MSA Layer는 Local Shifted Window 내에서 구현
  - Patch Embeddings는 더 깊은 Layer에서 합쳐짐
- test Dataset : **VTAB-1k**

주목할 점

- **Full Fine-Tuning** 대비, Specialized와 Structured는 약간 감소하였지만, Natural에서 +5.43% 향상
- **VPT** 대비, 0.02%의 파라미터를 추가 학습하면서 전체적인 성능 향상

Table 3: **Image classification accuracy for different pretrained objectives** — MAE [90] and MoCo v3 [26] with ViT-Base [23] as backbone. ⋆ denotes the rerun results that calibrate the VPT [4]

Pretrained objectives	MAE [90]				MoCo v3 [26]			
Methods	Tuned/ Total	VTAB-1k [78] [19] *Natural* [7]	*Specialized* [4]	*Structured* [8]	Tuned/ Total	VTAB-1k [78] [19] Natural [7]	Specialized [4]	Structured [8]
Full [CVPR22] [92]	100.00%	59.31%	79.68%	53.82%	100.00%	71.95%	84.72%	51.98%
Linear [CVPR22] [92]	0.04%	18.87% [0]	53.72% [0]	23.70% [0]	0.04%	67.46% [4]	81.08% [0]	30.33% [0]
Partial-1 [NeurIPS14] [93]	8.30%	**58.44%** [5]	**78.28%** [1]	47.64% [1]	8.30%	72.31% [5]	84.58% [2]	47.89% [1]
Bias [NeurIPS17] [30]	0.16%	54.55% [1]	75.68% [1]	**47.70%** [0]	0.16%	72.89% [3]	81.14% [0]	53.43% [4]
Adapter [NeurIPS20] [32]	0.87%	54.90% [3]	75.19% [1]	38.98% [0]	1.12%	74.19% [4]	82.66% [1]	47.69% [2]
VPT-S [ECCV22] [4]	0.05%	39.96% [1]	69.65% [0]	27.50% [0]	0.06%	67.34% [3]	82.26% [0]	37.55% [0]
VPT-D [ECCV22] [4]	⋆ 0.31%	36.02% [0]	60.61% [1]	26.57% [0]	⋆ 0.22%	70.27% [4]	83.04% [0]	42.38% [0]
GPT [ICML23] [101]	0.05%	47.61% [2]	76.86% [1]	36.80% [1]	0.06%	74.84% [4]	83.38% [1]	49.10% [3]
▶ Ours	0.38%	53.59% [6] {6}	77.75% [1] {3}	36.15% [1] {6}	0.22%	**77.47%** [5] {7}	**85.76%** [3] {4}	**58.74%** [6] {8}

- 백본 모델 : **ViT-Base**
- **MAE**와 **MoCO v3**에 대해서 실험 수행
- VFPT는 PEFT 기법들 중 가장 많은 승리
  - MAE : 8/19
  - MoCo v3 : 14/19

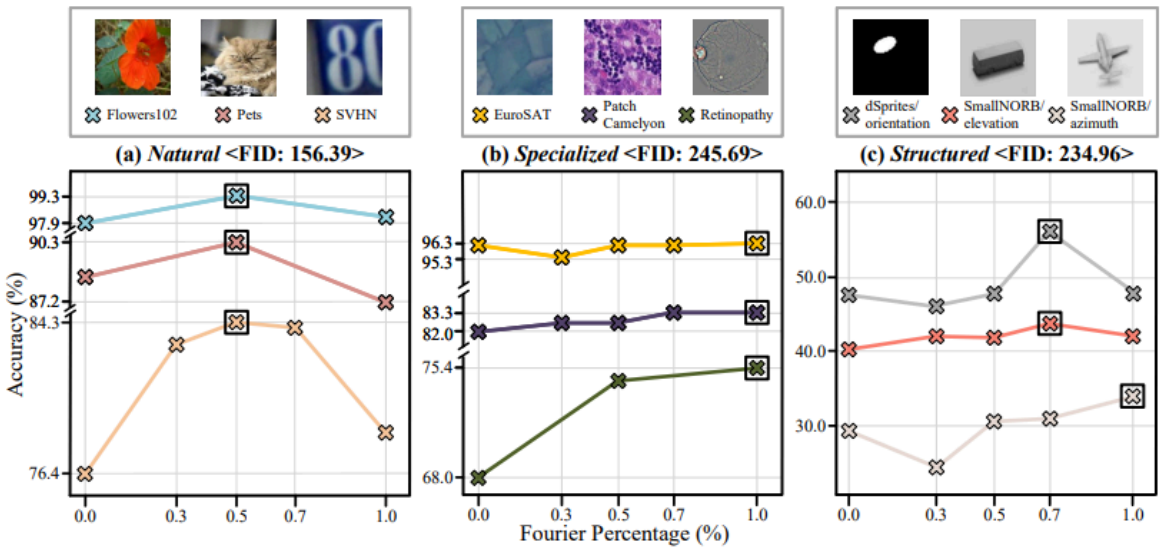


Figure 2: **Image classification accuracy of various Fourier percentages of VTAB-1k [78] for ViT-Base/16 [23].** For better illustration, we randomly select 3 datasets in each group of VTAB-1k. The "Average FID Score of Each Group" is reported in <·>. Our conclusion aligns with **16 of 19** cases. The cross framed by the square indicates the best percentage for each downstream task. Those datasets with only three Fourier percentage reports are due to the prompt length limits.

- VFPT 내에서 **Fourier Prompt의 비율 조정**하며 결과 분석

Natural 그룹

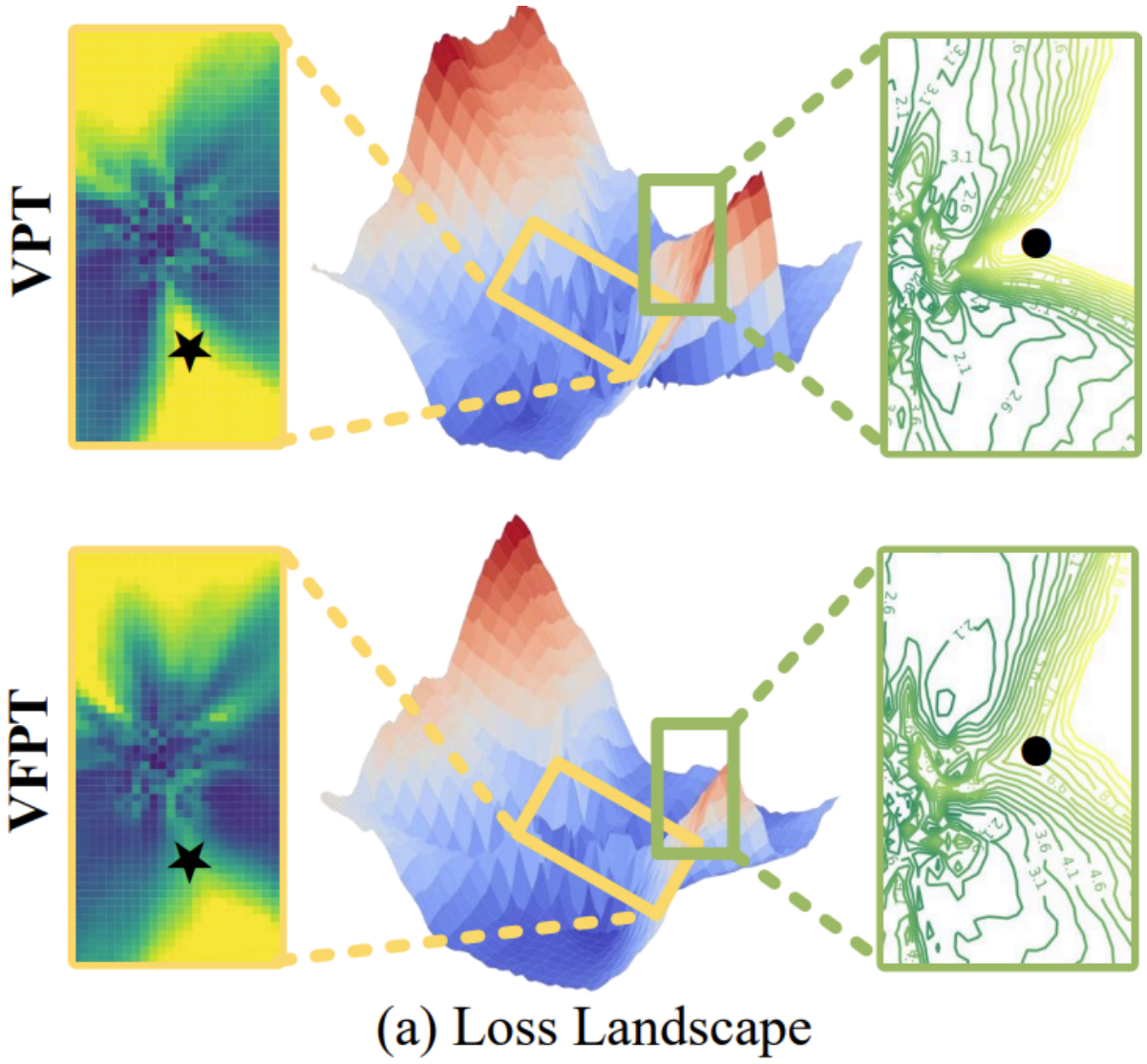
- ImageNet21K와 분포 유사
- Visual Prompt의 절반만 Fourier로 변환했을 때 최고 성능

Specilaized & Structured 그룹

- 분포 차이가 커서 transfer learning 어려움
- **Fourier 비율을 높일수록 정확도 증가**

4.3. Study of Optimization

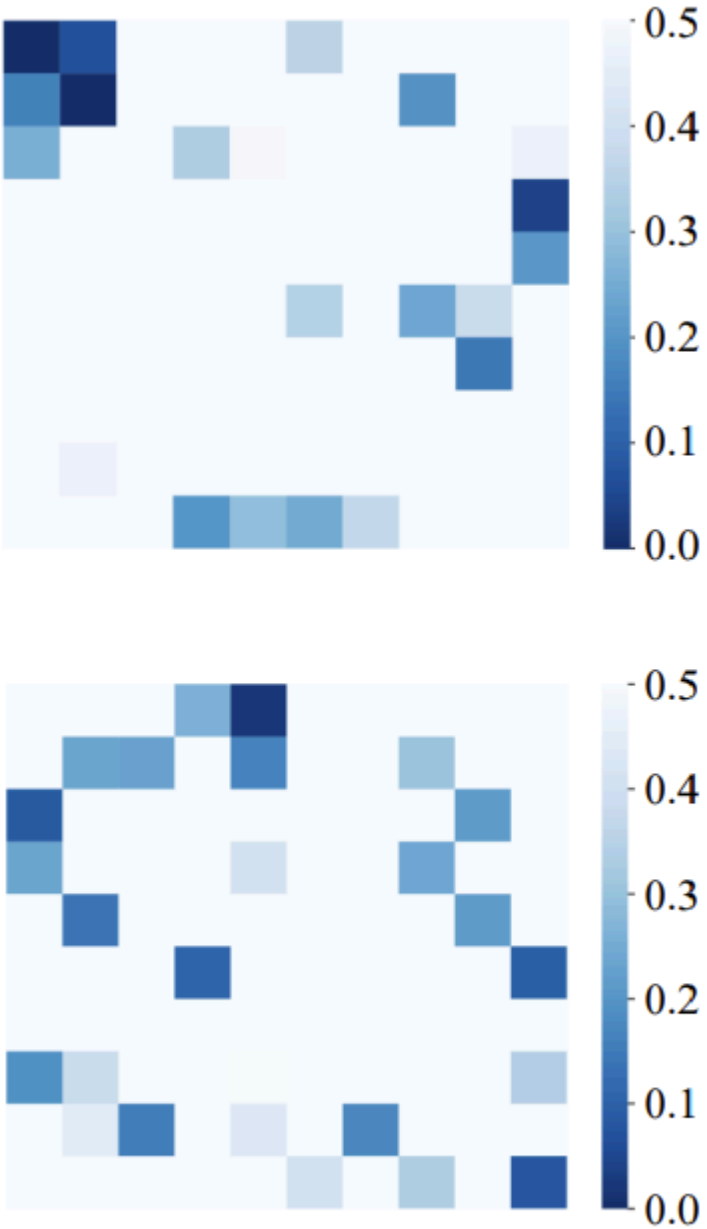
- **최적화 관점에서의 VFPT**
  - 이전 연구에 따르면, **Loss Landscape의 기하학적 특성이 모델의 일반화 성능에 큰 영향**을 준다고 함
  - **Loss Landscape**와 **Hessian Matrix**를 분석



(a) Loss Landscape

1. **Flatness**
    - **Loss Landscape**
        - 모델 파라미터 공간에서 **Loss 값이 어떻게 변하는지** 나타낸 지형
        - Flat 하면, 조금만 움직여도 손실이 크게 변하지 않음. → 일반화 성능 좋음
    - 최종적으로 VFPT는 **평탄한 최적점을 유도** → 더 낮은 test error , 더 좋은 Generalization
2. **Convexity**
    - 손실 함수의 Hessian Matrix의 고유값을 보면 지형이 얼마나 볼록하거나 평평한지 알 수 있음
        - 고유값 > 0 : Convex
        - 고유값 = 0 : Flat

    - 고유값 < 0 : Concave
- VFPT가 0 근처의 값이 더 많다.

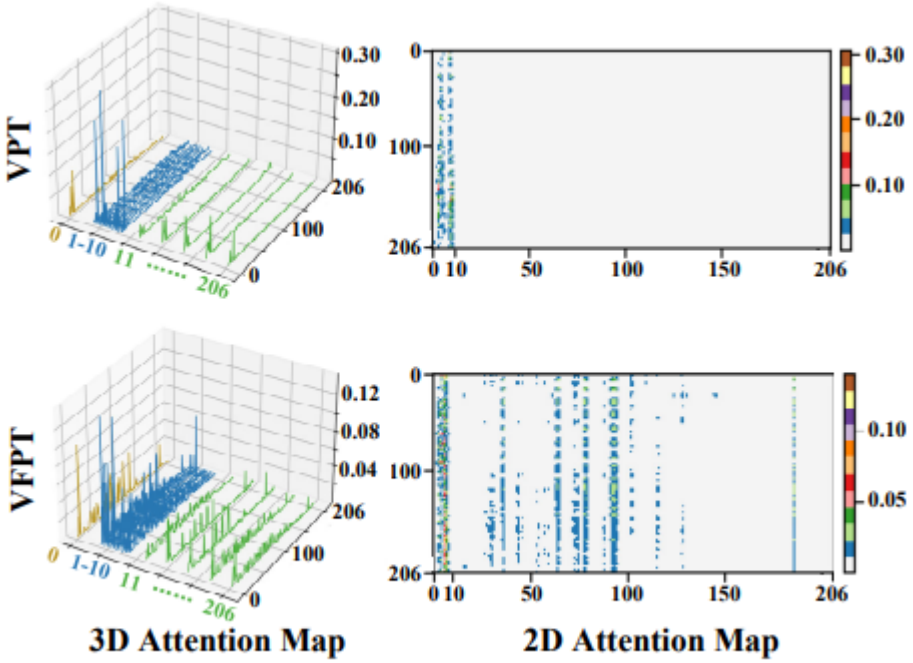


(b) Ratio Map of Hessian

4.4. Study of Interpretability

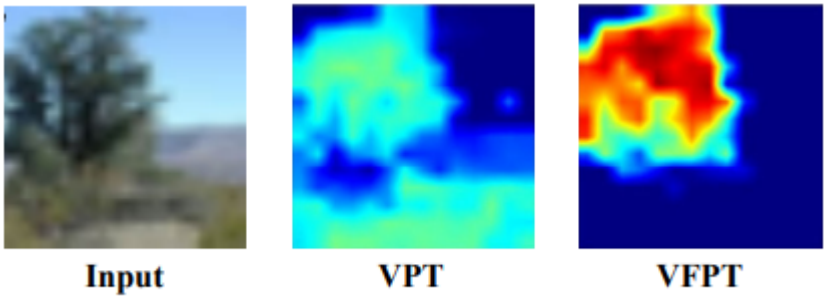
> 기존의 Prompt Tuning은 학습된 Prompt의 역할 혹은 의미를 명확하게 해석하기 어려움
> VFPT에서는 이를 해결하기 위해 **시각화 및 정성적 분석 수행**

- **VPT와 VFPT의 Attention Map (3D, 2D) 비교**



(a) Attention map

- **Grad-CAM을 이용한 시각화**



(b) GradCAM Heatmap

1. **Prompt와 Attention**
    - VPT와 VFPT 모두 **Prompt 위치에 강한 Attention** → **Fine-tuning 중 Prompt가 Frozen Embedding에 강한 영향**
2. **VFPT의 Global Attention pattern**

- VFPT는 VPT보다 Attention이 넓고 강하게 퍼짐
- Fourier Prompt가 전체 Transformer 입력 공간과 강하게 상관됨

3. **Grad-CAM**
   - VFPT는 **foreground**와 **background**를 명확하게 분리

4.5. Ablation Study

🔷 Transform Type

Table 4: **Ablative studies of transform type** on VTAB-1k [78] *Natural* and *Specialized* benchmarks in three runs. Per-task results are available in Appendix.

Transform Type (Domain)	Transform Dimension Sequence	Hidden	VTAB-1k [78] [19] *Natural* [7]	*Specialized* [4]
FLL ($\mathcal{S}$)		✓	80.98%	84.02%
LLL ($\mathcal{S}$)		✓	80.54%	82.64%
FFT ($\mathcal{F}$) + FDA ($\mathcal{F}$) [71]	✓	✓	80.90%	84.03%
FFT ($\mathcal{F}$)	✓	✓	**81.35%**	**84.93%**

- FLL : Fixed Linear Layer
- LLL : Learnable Linear Layer
- FFT + FDA :

🔷 Fourier Prompt Dimension

- 논문 저자들은 **Sequence Length**와 **Hidden Dimension**에 모두 **2D FFT** 적용
- 각각 적용 시의 결과 확인

Fourier Dimension		VTAB-1k [78] [19]	
Sequence	Hidden	*Natural* [7]	*Specialized* [4]
✓		80.88%	83.57%
	✓	80.74%	83.87%
✓	✓	**81.35%**	**84.93%**

(a) Fourier Prompt Dimension

🔷 Fourier Prompt Location

- Fourier Prompt의 삽입 위치를 변경해가며 삽입
- **Prepend($\mathcal{P}$), Append($\mathcal{A}$), Random($\mathcal{R}$)**

Prompt Location	VTAB-1k [78] [19]	
	Natural [7]	*Specialized* [4]
$\mathcal{A}$	81.02%	83.80%
$\mathcal{R}$	78.62%	82.47%
$\mathcal{P}$	**81.35%**	**84.93%**

(b) Fourier Prompt Location

🔷 Fourier Prompt Depth

- Prompt를 몇 개의 Layer에 넣을 지 결정

Prompt Depth	VTAB-1k [78] [19] *Natural* [7]	*Specialized* [4]
1 3 5 7 9 11	80.48%	83.73%
1-6	80.79%	84.34%
7-12	80.83%	83.93%
1-12	**81.35%**	**84.93%**

(c) Fourier Prompt Depth

5. Conclusion

1. Fourier Transform을 활용한 **VFPT** 소개
2. 직관적이면서도 효과적인 설계를 통해 **Spatial 정보와 Frequency 정보를 통합**
3. 다양한 데이터셋 간의 일반성을 입증