2-1 나비 데이터 학습 진행

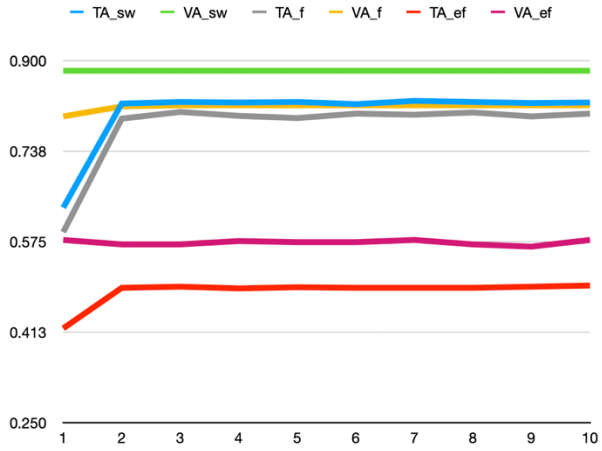


모델	Optimizer	Loss	lr	batch
Swin_t	Nadam	LabelSmoothing	0.001	128
Swin_t	Adam	LabelSmoothing	0.001	128
EfficientnetB4	Nadam	LabelSmoothing	0.001	64
EfficientnetB4	Adam	LabelSmoothing	0.001	64
facebook	Nadam	LabelSmoothing	0.001	340
facebook	Adam	LabelSmoothing	0.001	340

2-2 학습 결과

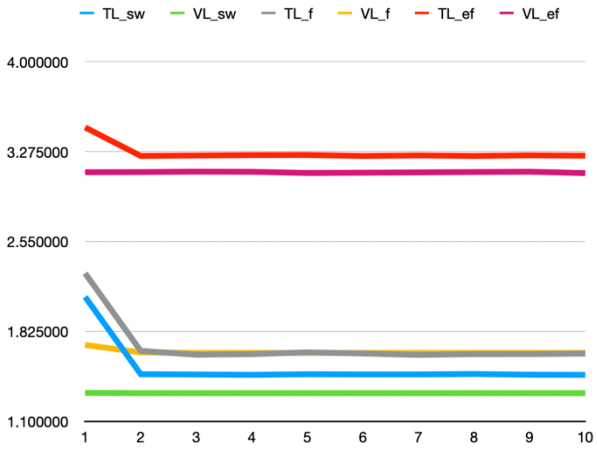
나비 Adam Accuracy

Ep	TA_sw	VA_sw	TA_f	VA_f	TA_ef	VA_ef
1	0.636	0.882	0.592	0.8	0.419	0.578
2	0.823	0.882	0.796	0.818	0.492	0.57
3	0.826	0.882	0.808	0.82	0.494	0.57
4	0.825	0.882	0.801	0.82	0.491	0.576
5	0.826	0.882	0.797	0.82	0.493	0.574
6	0.822	0.882	0.805	0.82	0.492	0.574
7	0.828	0.882	0.803	0.82	0.492	0.578
8	0.826	0.882	0.807	0.82	0.492	0.57
9	0.824	0.882	0.8	0.82	0.494	0.566
10	0.825	0.882	0.805	0.82	0.496	0.578



나비 Adam Loss

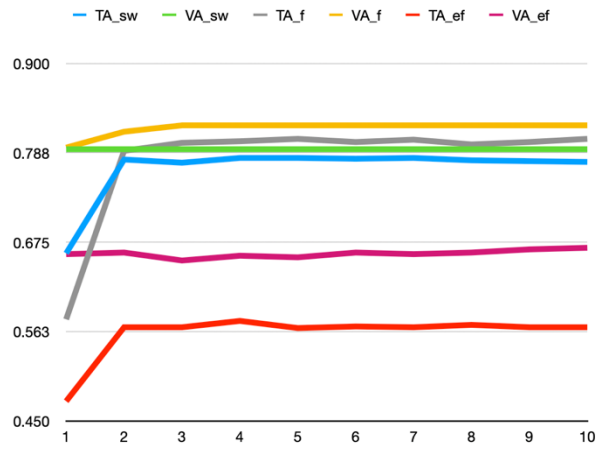
Ep	TL_sw	VL_sw	TL_f	VL_f	TL_ef	VL_ef
1	2.103958	1.329989	2.293892	1.716454	3.470463	3.109188
2	1.480998	1.327789	1.66866	1.656331	3.240352	3.110602
3	1.478297	1.327789	1.637809	1.652422	3.243511	3.113875
4	1.475268	1.327789	1.642588	1.652245	3.247006	3.11255
5	1.480635	1.327789	1.656746	1.652238	3.247671	3.103191
6	1.478767	1.327789	1.647662	1.652238	3.240087	3.105307
7	1.479072	1.327789	1.636004	1.652238	3.243991	3.107825
8	1.482449	1.327789	1.642847	1.652238	3.240222	3.110878
9	1.47639	1.327789	1.642993	1.652238	3.244888	3.11317
10	1.475052	1.327789	1.646623	1.652238	3.241932	3.10221



swin과 facebook 모델은 비교적 괜찮은 정확도와 낮은 Loss를 보여줬으나 Efficientnet 모델은 다소 낮은 정확도와 높은 Loss를 보여줌

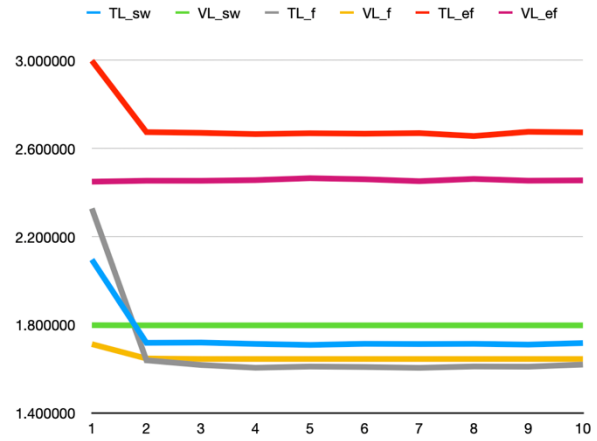
나비 Nadam Accuracy

Ep	TA_sw	VA_sw	TA_f	VA_f	TA_ef	VA_ef
1	0.661	0.792	0.578	0.794	0.475	0.66
2	0.779	0.792	0.79	0.814	0.568	0.662
3	0.775	0.792	0.8	0.822	0.568	0.652
4	0.781	0.792	0.802	0.822	0.576	0.658
5	0.781	0.792	0.805	0.822	0.567	0.656
6	0.78	0.792	0.801	0.822	0.569	0.662
7	0.781	0.792	0.804	0.822	0.568	0.66
8	0.778	0.792	0.798	0.822	0.571	0.662
9	0.777	0.792	0.801	0.822	0.568	0.666
10	0.776	0.792	0.805	0.822	0.568	0.668



나비 Nadam Loss

Ep	TL_sw	VL_sw	TL_f	VL_f	TL_ef	VL_ef
1	2.096203	1.79846	2.328044	1.712495	2.998562	2.449605
2	1.718468	1.797699	1.639186	1.64735	2.674467	2.453918
3	1.719753	1.797699	1.617857	1.645179	2.671233	2.453653
4	1.713349	1.797699	1.60542	1.645061	2.665472	2.456695
5	1.708861	1.797699	1.610844	1.645057	2.669461	2.465341
6	1.713876	1.797699	1.608685	1.645057	2.667352	2.460511
7	1.712725	1.797699	1.605029	1.645057	2.670126	2.451899
8	1.7138	1.797699	1.611147	1.645057	2.656634	2.462001
9	1.710314	1.797699	1.610213	1.645057	2.675953	2.454105
10	1.717414	1.797699	1.619728	1.645057	2.673256	2.455578



Optimizer를 Nadam으로 교체한 뒤에도 유독 Efficientnet 모델만 좋지 않은 결과를 보여줌

2-3 테스트 결과

```
C:\Miniconda\envs\molru\python.exe "C:/Users/labadmin/Desktop/New folder/0109/test.py"
Swin_Adam Acc >> 0.902

Process finished with exit code 0
```

모델 Swin Optimizer Adam 테스트 결과 정확도 0.9 수준의 준수한 성능을 보여줌

```
C:\Miniconda\envs\molru\python.exe "C:/Users/labadmin/Desktop/New folder/0109/test.py"
C:\Miniconda\envs\molru\lib\site-packages\torchvision\models\_utils.py:208: UserWarning:
  warnings.warn(
C:\Miniconda\envs\molru\lib\site-packages\torchvision\models\_utils.py:223: UserWarning:
  warnings.warn(msg)
Effi_Adam Acc >> 0.548

Process finished with exit code 0
```

모델 Efficientnet Optimizer Adam 테스트 결과 학습 때와 마찬가지로 좋지 않은 성능을 보여줌