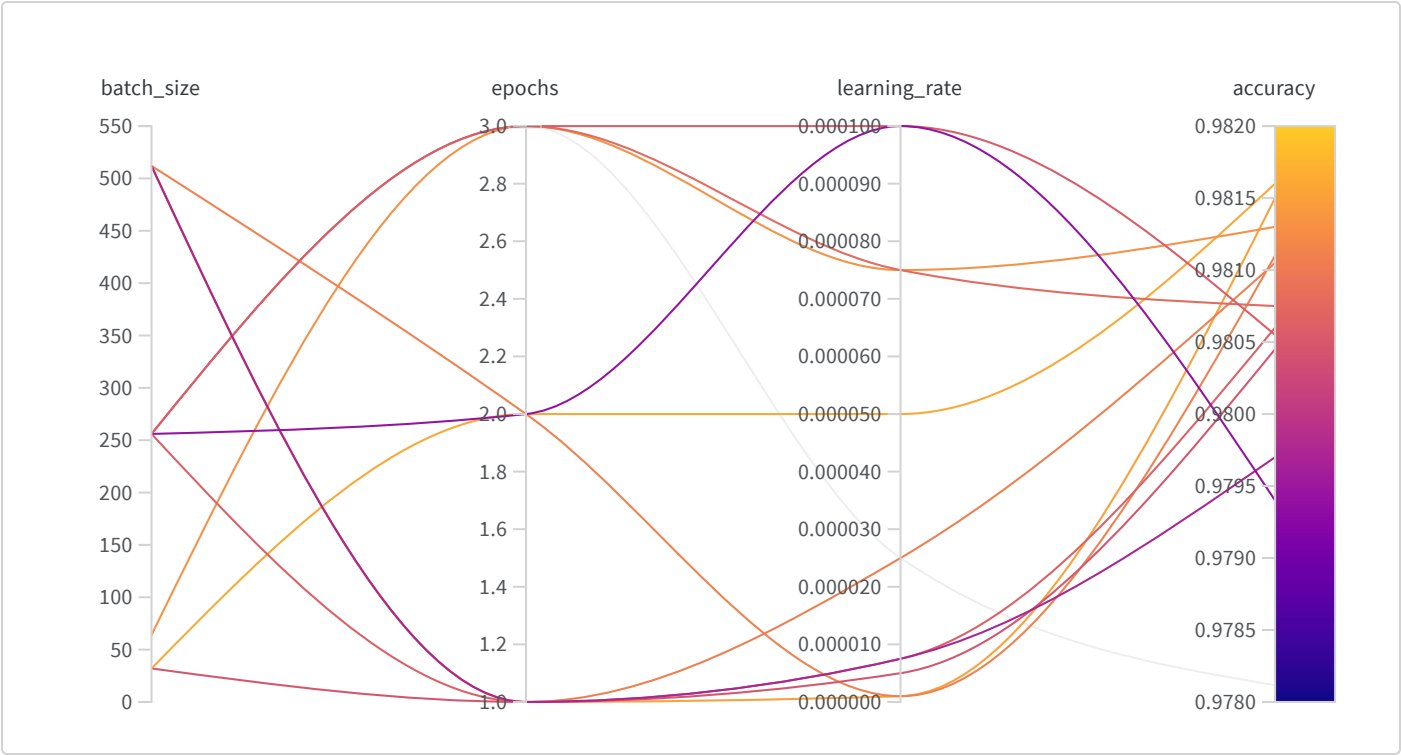
PROJECT 1 _ TEXT CLASSIFICATION

using BERT

Jeonhyotaek

Type '/' for commands

Training 1



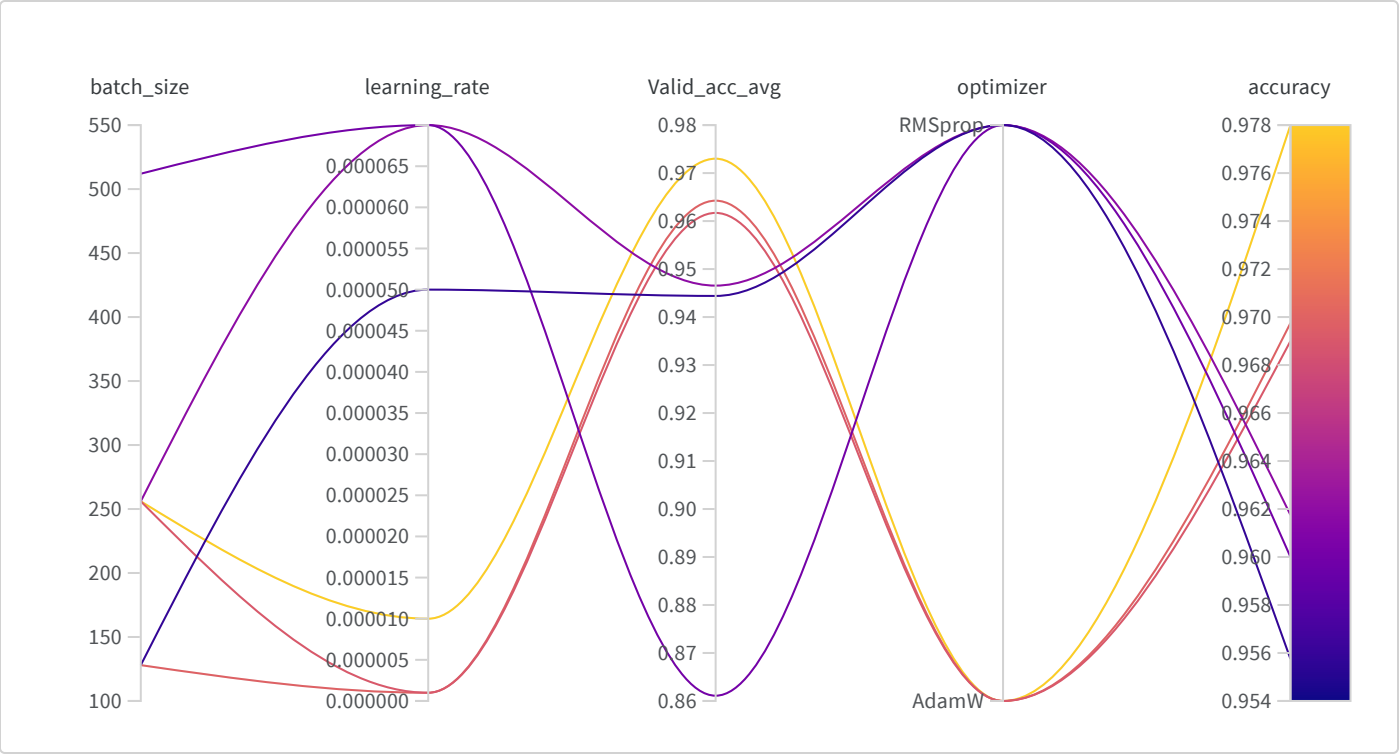
Import panel Add panel

batch size : 32~512
epoch : 1~3
learning rate : 1e-4 ~ 1e-6

Try to find hyperparameters that maximize validation accuracy.
The problem with the first attempt was that the traing & validation batch size were not considered

> together
> The encouraging thing is that when the learning_rate is less than 1e-5, the validation accuracy is usually high.
> it would be easier to find other parameters to reduce the epoch value and apply it to various test cases.
> 한줄요약 : 배치사이즈를 하나를 고정으로 두고 학습 진행. 제대로 학습 X. learning_rate에 대해 유의미한 값 얻음.

Training 2

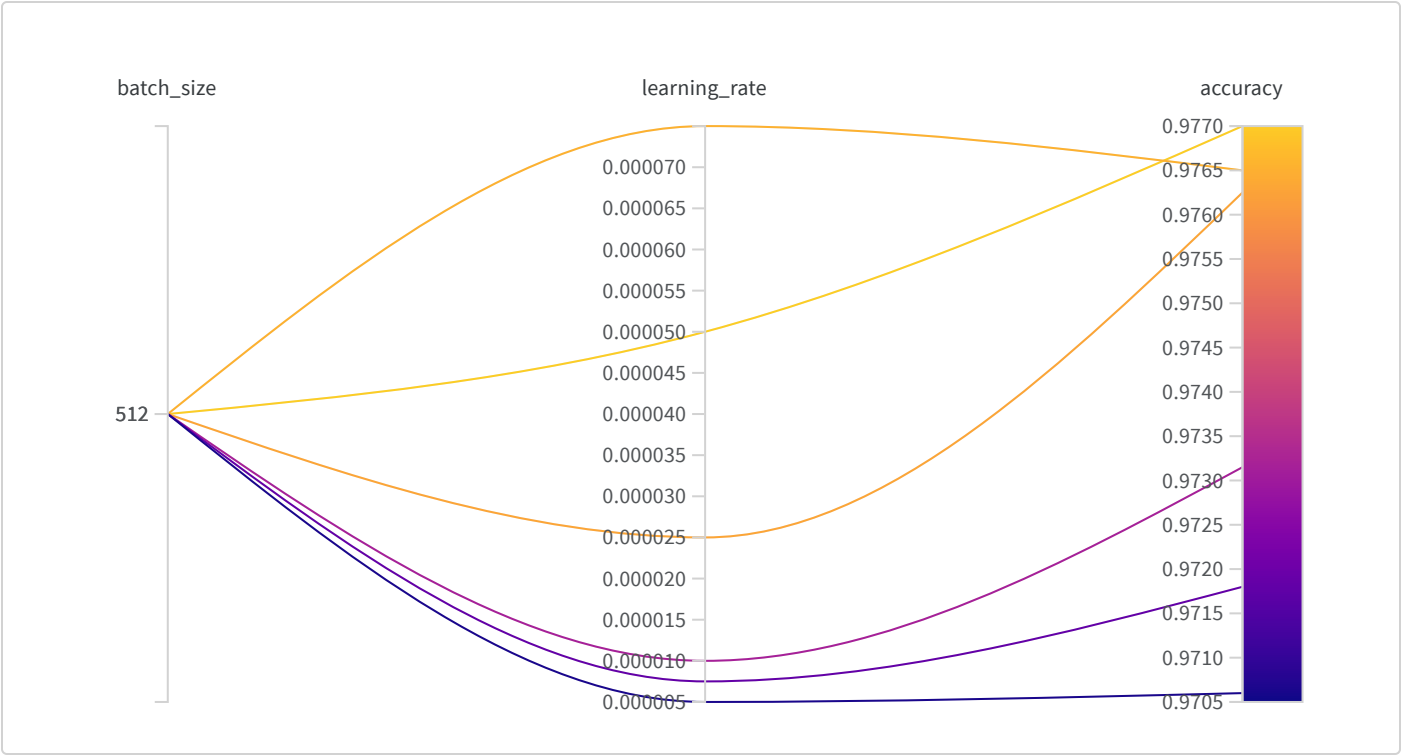


Import panel Add panel

batch size : 32~512
epoch : 1
learning rate : 1e-5 ~ 1e-6

Before proceeding with training, I wanted to apply something else to optimizer.
Therefore, AdamW and RMSprop were put together in the test case.
And the learning rate was designated between 1e-5 and 1e-6 through the result value checked above.
As a result, the difference between RMSprop and AdamW is clearly large,
so the experiment was stopped early, and AdamW will be used for next training.
한줄요약 : optimizer를 다양한 모델로 사용해보고 싶었으나 AdamW의 성능이 좋아 AdamW로 다음실험 진행.

Training 3



Import panel Add panel

Training 1에서 얻은 결과값을 토대로 5자리수 이하의 learning rate 중에서 어느 범위의 learning rate가 효과적일지에 대한 판단을 위해 1e-4 ~ 1e-6 범위의 learning rate의 학습을 진행한 결과 5e-5 ~ 7.5e-5

사이의 값이 최적의 learning_rate임을 알 수 있음.

Import panel Add panel

validation_acc

Draft autosaved just now Save

Import panel Add panel

epoch 10으로 두고 학습한 결과. 1~5까지는 유의미한 결과를 얻을 수 있으나 그 이후부터는 변화가 미비함. 다른 모델들에서도 적용했을때 대체적으로 epoch는 3~5 사이의 값으로 두는게 가장 합리적임.

<hyperparameter 조정 결과>

-> learning_rate : 5e-5 ~ 7.5e-5 // batch size에 비례해서 조절해야함. batch가 커지면 learning rate도 함께 커야하고, batch가 작으면 함께 작은 값으로 학습해야함.
-> 각 hyperparameter 튜닝 시에는 나머지 값들 전부 통일 시켜둔 뒤에 진행.

-> (1) **optimizer** : **AdamW**가 가장 높은 성능
: optimizer 세부 조정 (weight decay, eps) 설정 [0.9727]
(1-1) 미세조정 **weight decay** 1e-4 ~ 1e-7(1e-6), **eps** 1e-2 ~ 1e-4 (1e-2) ; weight decay는 1e-6, eps는 1e-2에서 가장 높은 성능을 보임.
(1-2) weight decay 1e-6 / eps 1e-2 에서 조금씩 조정.
(1-3) 추가적인 optimizer = BERTAdam으로 진행 했을때 큰 차이 없음.

-> (2) **batch_size** 최대 조정 (train 256 / valid 512) [0.9770] (batch_size는 최대로)
: batch_size : [32,64,128,256,512] 중 최대 batch_size가 가장 높은 성능을 보임.
(2-1) train = valid batch_size [0.9760] <-> train 128 valid 512 [0.9769] ; 유의미한 차이는 없음.
(2-2) batch size = 1000으로 지정했을 때는 큰 차이 없음.

-> (3) **epoch** 1,3 값 진행. (epoch을 늘려도 lr 값에 대한 순위 변동이 있는지 확인 -> epoch 3 이후의 training loss 추적하며 판단. 값이 계속수렴하여 변동이 크게 없을 때는 학습 종료. ~>)
: epoch 값이 3까지 커질 때 당연하게도 성능이 높아지지만 5 이상의 값들에 대해서는 유의미한 향상을 볼 수 없었음.

-> (4) Learning rate : 5e-5 주변의 learning rate값이 성능이 높았고, learning rate scheduler를 사용해서 성능을 평가해봄
: LambdaLR, StepLR, 등을 사용해 보았을 때 성능이 초반에는 큰 영향이 없다가 후반부의 loss 값들이 떨어지는것을 확인할 수 있었음. learning rate scheduler를 사용하여 epoch마다 lr값 수정하도록 적용.

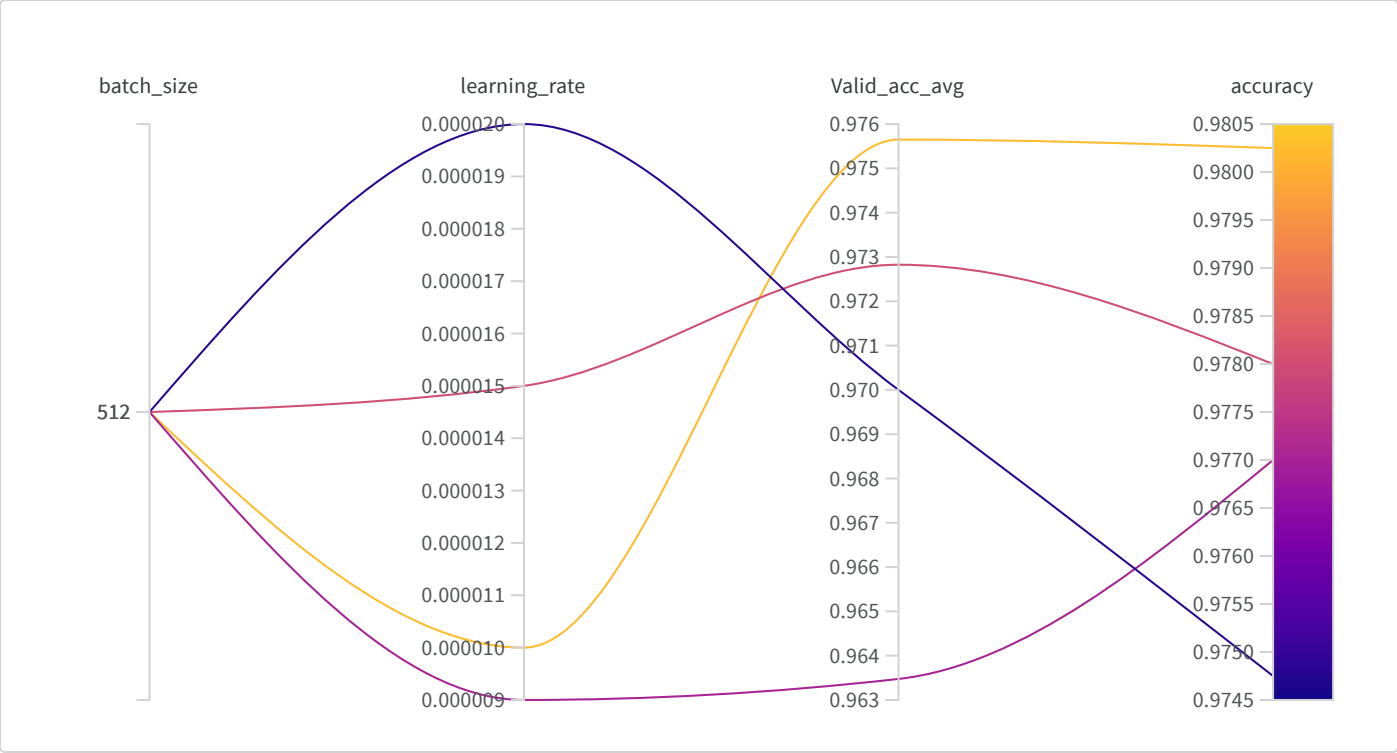
[결론] : hyperparameter를 다방면으로 조절해가며 loss와 acc값을 확인해보았는데 batch_size 이외에는 유의미한 결과값을 얻지 못했다. 성능을 높이기 위해 다른 모델들도 검토하는 방향으로 실험을 설계했다. 모델 성능이 크게 이상없는한 train 과정에 포함시켜 voting 기법을 적용해 학습 정확도를 높이는 방향으로 진행한다.

Different type of BERT Models

albert-base-v2

기본 코드 accuracy : Acc for model which have lower valid loss: 0.97775

lr scheduler / batch_size 조정 후 learning_rate <-> accuracy 결과 1e-5 최대값. (이외의 값들 더 해봤으나 의미없는 값으로 나와 제거.)

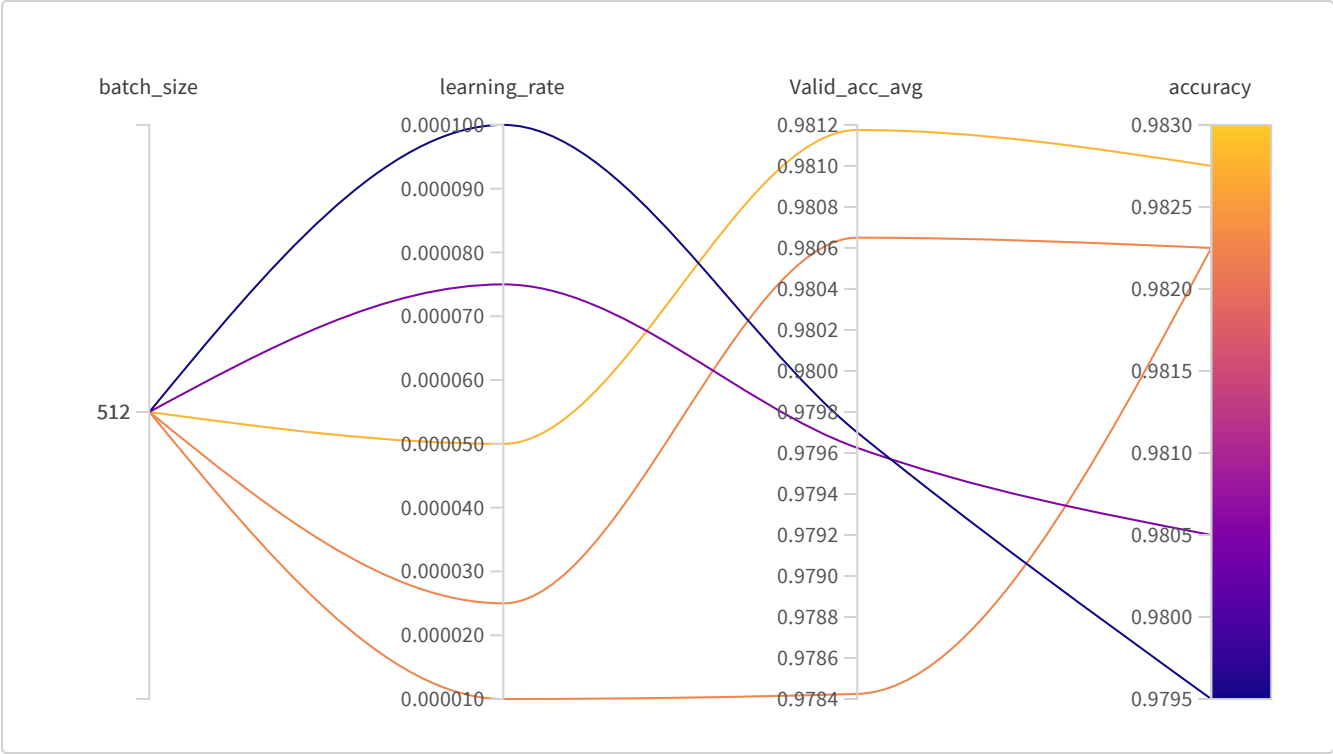


Import panel Add panel

Electra

기본 코드 accuracy : Acc for model which have lower valid loss: 0.9820

lr scheduler / batch_size 조정 후 learning_rate <-> accuracy 결과 5e-5 최대값.

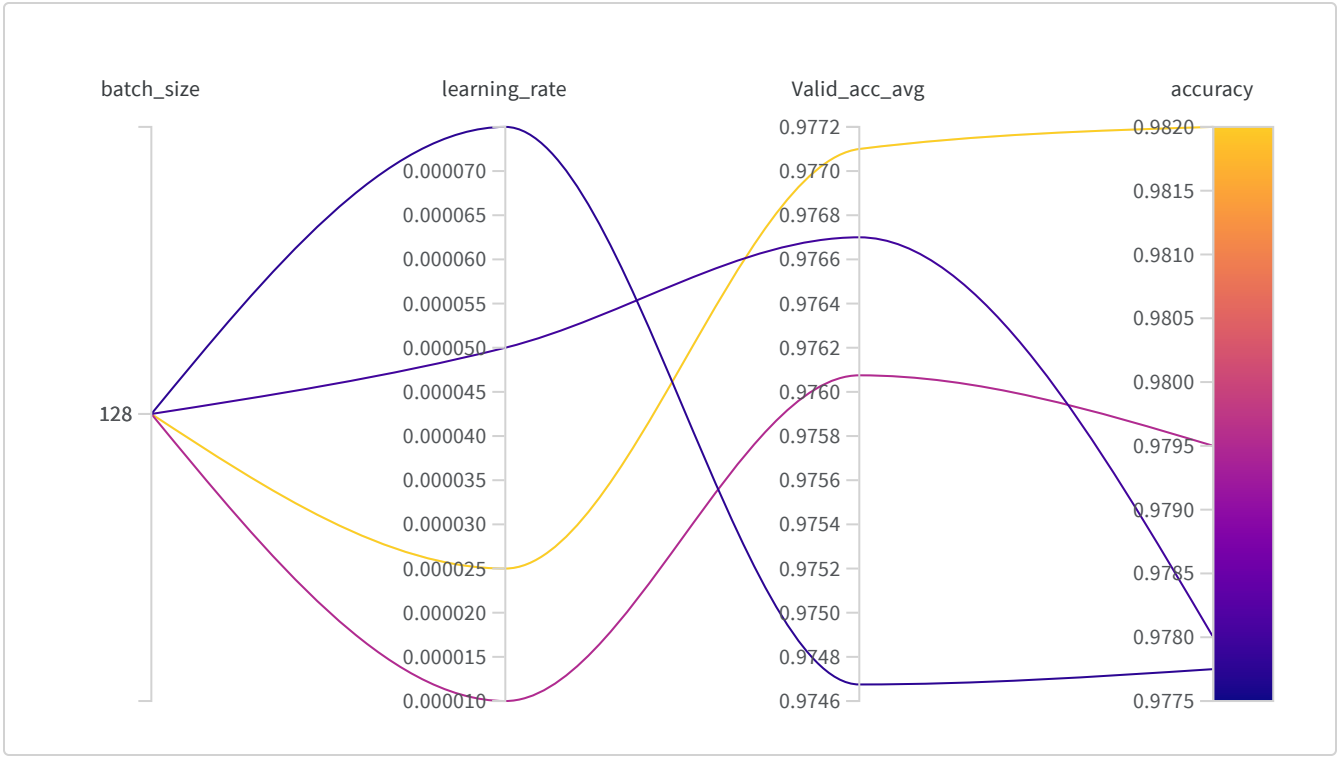


Import panel Add panel

RoBERTa

기본 코드 accuracy : Acc for model which have lower valid loss: 9810

lr scheduler / batch_size 조정 후 learning_rate <-> accuracy 결과



Import panel Add panel