WELCOME TO THE

Molecular Team Lecture Series

In this lecture series, MAI LAB Molecular Team will introduce various molecular generation tasks

TODAY'S LECTURE

Improving Generalization in Meta-learning via Task Augmentation

0 Abstract

Abstract

- Meta learning 은 novel task 를 learning 하는데에 있어서 previous knowledge 를 가지고 오는 효과적인 방법임
- 가장 유행하는 Meta Learning 방법으로는 support set 을 이용한 model initialization 이 있음
  - 이것의 핵심은 query set 에 대한 model 의 performance 를 측정하는 것임
- 하지만 이 방법은 meta-training task 에 overfit 될 수 있다는 단점이 있음

Abstract

- 이것을 해결하기 위해 data를 더 많이 (more data) 이용해서 meta-training task 를 augment 하는 방법을 사용함
- 2가지의 task augmentation method 를 사용했는데,
  - MetaMix - Support set 과 Query set 두가지로 부터 얻은 샘플의 feature 와 label 을 선형 결합
  - Channel Shuffle - 서로 다른 class 에 대응하게끔 channel 의 subset 을 random 하게 replace 함

MT

1 Introduction

About Meta-learning

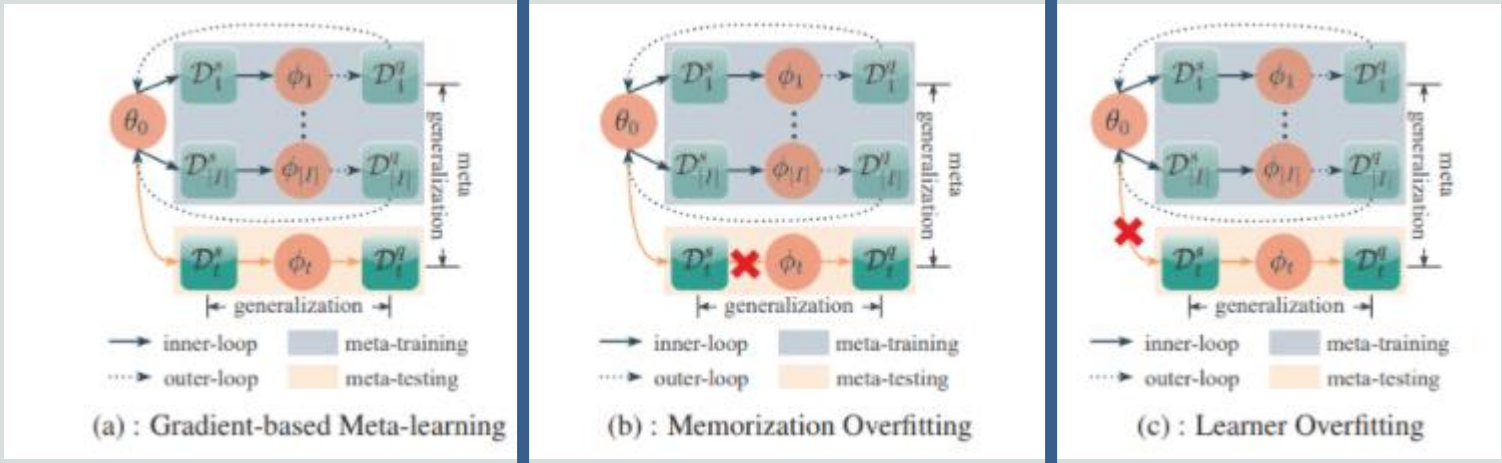
- Meta-learning, or learning to learn, empowers agents with the core aspect of intelligence—quickly learning a new task with as little as a few examples by drawing upon the knowledge learned from prior tasks

- Some of the dominant algorithms learn a transferable metric space from previous tasks
  - Unfortunately being only applicable to classification problems

- Instead, gradient-based algorithms (Finn et al., 2017; 2018) framing meta-learning as a bi-level optimization problem are flexible and general enough to be independent of problem types

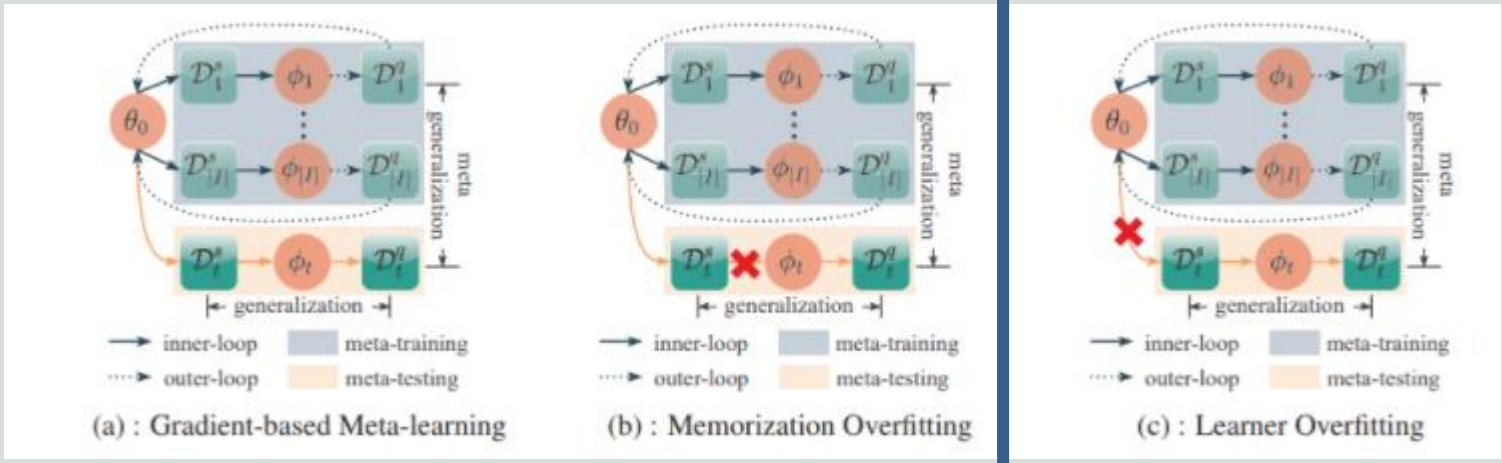
Learned initialization 은 2가지 문제를 가지고 있음

- **Memorization overfitting**
  - ✓ 원래는 support set 에 의해 fine tuning 이 되어야 하는데 그러지를 못함!!



Learned initialization 은 2가지 문제를 가지고 있음

- **Learner overfitting**
  - ✓ Meta training data에 overfitting 이 되어서 meta test 가 아예 불가능함
  - ✓ 심지어 support set 이 열심히 fine tuning 을 하기위해 몸을 바치지만 그래도 불가능함



이 문제를 해결하기 위해,

- The few existing solutions attempt to regularize the search space of the initialization
- Rather than passively imposing regularization on the initialization
  - (Initialization 에 의존하지 않고 다른 방식에 의존)
- Active data augmentation 방법 또한 존재. 즉, support 및 query set 에 동일한 노이즈를 주어 더 많은 data 를 meta train 에 이용하였다
  - Therefore, little extra knowledge is introduced to meta-train the initialization

This paper,

1. Task augmentation 을 통해 더 많은 data를 생성하는 flexible 하고 powerful 한 방법을 제안
2. Task augmentation 의 목표는 support set 에 대한 target prediction 의 의존성을 높여주고 모델의 initialization 을 최적화하는 추가적인 지식을 제공한다

This paper,

1. MetaMix, Channel Shuffle
    - MetaMix : Support set 과 Query set 의 original feature 와 Neural Network의 hidden representation 을 선형결합한 후, 이에 해당하는 label 에 같은 선형 보간법을 적용시킨다
    - 분류 문제에 대해서, MetaMix 는 Channel Shuffle 이라는 방법에 의해 enhance 되는데, 이는 각각의 class 에 대해 channel 의 subset 을 다른 class 에 해당하는 sample 과 replace 하는 역할을 함으로써 진행된다
2. These additional signals for the meta-training objective improve the meta-generalization of the learned initialization as expected

2 Preliminaries

Brief explanation of meta-learning

- Support set 은 다음과 같이 표현 가능

$$\mathcal{D}_i^s = \{(\mathbf{x}_{i,j}^s, \mathbf{y}_{i,j}^s)\}_{j=1}^{K^s}$$

  - i 번째 task 에서 x1, y1 ..... xK yK 까지 데이터들이 존재 (s 는 support set)

- Query set 도 마찬가지

- 아래와 같이 bi-level optimization problem 과 같은 문제를 해결 할 수 있다

$$\theta_0^* := \min_{\theta_0} \mathbb{E}_{\mathcal{T}_i \sim p(\mathcal{T})} \left[\mathcal{L}(f_{\phi_i}(\mathbf{X}_i^q), \mathbf{Y}_i^q) \right],$$

$$\text{s.t. } \phi_i = \theta_0 - \mu \nabla_{\theta_0} \mathcal{L}(f_{\theta_0}(\mathbf{X}_i^s), \mathbf{Y}_i^s),$$

3 Task Augmentation

Task Augmentation

1. 다음의 수식을 생각해보면

$$\theta_0^* := \min_{\theta_0} \frac{1}{n_T} \sum_{i=1}^{n_T} \left[\mathcal{L}(f_{\phi_i}(\mathbf{X}_i^q), \mathbf{Y}_i^q) \right],$$

$$\text{s.t. } \phi_i = \theta_0 - \mu \nabla_{\theta_0} \mathcal{L}(f_{\theta_0}(\mathbf{X}_i^s), \mathbf{Y}_i^s)$$

즉, fir
는 뜻

2. Ove
을 ㅊ
   -
   -

3. Dat

❖ Mix-up 알고리즘 동작 원리

- 두 데이터 샘플로부터 선형 보간법을 통해 새로운 샘플을 생성하는 데이터 증강 기법
- 간단한 동작 원리에도 좋은 일반화 성능을 보장하여 다양한 딥러닝 연구 분야에서 사용되고 있는 기법

Mix-up formulation

$$\tilde{x} = \lambda x_i + (1-\lambda) x_j, \quad \text{where } x_i, x_j \text{ are raw input vectors}$$
$$\tilde{y} = \lambda y_i + (1-\lambda) y_j, \quad \text{where } y_i, y_j \text{ are one-hot label encodings}$$

$$\lambda \sim Beta(\alpha, \alpha)$$

Definition

1. Augmentation function을 g(.) 라고 하자. 그렇다면 augmented function 은 dataset 단에서 아래와 같이 변형 가능하다

$$\mathcal{T}_i = \{\mathcal{D}_i^s, \mathcal{D}_i^q\} \Rightarrow \mathcal{T}_i^{'} = \{g(\mathcal{D}_i^s), g(\mathcal{D}_i^q)\}$$

2. 이와 같은 경우 2개의 crietria 를 얻을 수 있는데 아래와 같음

$$(1)\ I(g(\hat{\mathbf{Y}}_i^q); g(\mathcal{D}_i^s)|\theta_0, g(\mathbf{X}_i^q)) - I(\hat{\mathbf{Y}}_i^q; \mathcal{D}_i^s|\theta_0, \mathbf{X}_i^q) > 0,$$

$$(2)\ I(\theta_0; g(\mathcal{D}_i^q)|\mathcal{D}_i^q) > 0.$$

Definition

2. 이와 같은 경우 2개의 criteria 를 얻을 수 있는데 아래와 같음

$$(1)\ I(g(\hat{\mathbf{Y}}_i^q); g(\mathcal{D}_i^s)|\theta_0, g(\mathbf{X}_i^q)) - I(\hat{\mathbf{Y}}_i^q; \mathcal{D}_i^s|\theta_0, \mathbf{X}_i^q) > 0,$$

$$(2)\ I(\theta_0; g(\mathcal{D}_i^q)|\mathcal{D}_i^q) > 0.$$

1)

- Memorization Overfitting 을 극복하기 위한 방법임
- 모델이 Support set 에 좀 더 의존 할 수 있도록 만들어주는 것
- g(Yq) 와 g(Ds) 와의 mutual information 을 키우자
- inner 단에서, X query 와 theta_0 가 주어졌을 때, Support set 과 Y query 의 mutual information 보다 augmentation 한 상태가 더 크다는 것을 의미

Definition

2. 이와 같은 경우 2개의 crietria 를 얻을 수 있는데 아래와 같음

$$(1)\ I(g(\hat{\mathbf{Y}}_i^q); g(\mathcal{D}_i^s)|\theta_0, g(\mathbf{X}_i^q)) - I(\hat{\mathbf{Y}}_i^q; \mathcal{D}_i^s|\theta_0, \mathbf{X}_i^q) > 0,$$

$$(2)\ I(\theta_0; g(\mathcal{D}_i^q)|\mathcal{D}_i^q) > 0.$$

2)

- Learner Overfitting (Outer loop) 와 관련있음
- Augmented(Query set) 이 outer-loop 에서 initialization 에 더 큰 기여를 하게끔 하자는 것

1. MetaMix

- 먼저, 가장 심플하게 augmentation 을 하는 방법은 support set 을 outer loop 에 사용하는 방법일 것이다. 하지만 이 방법은...좀 구리다. 이미 Support set 을 통해서 task 별로 최적화된 adapted model 의 퍼포먼스에 의해 측정되기 때문

- 그리하여, More data 의 방법을 어떻게 하였느냐면, 이용할 수 있는 support set 과 query set 에 MetaMix 를 적용하기로 하였다.
  - Mix-up 을 사용하였는데, 관련 논문은 아래와 같다
  - Mix-up 에 관련된 논문
    - https://openreview.net/pdf?id=r1Ddp1-Rb
  - MetaMix 논문
    - http://proceedings.mlr.press/v97/verma19a/verma19a.pdf

Mix-up

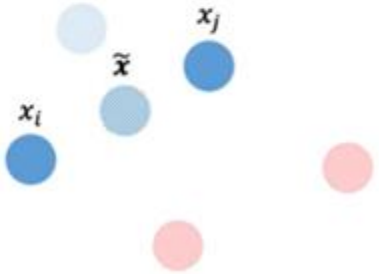
❖ Mix-up 알고리즘 동작 원리

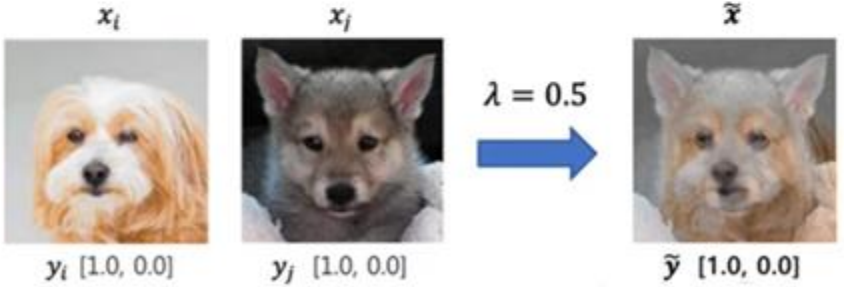
- 두 데이터 샘플로부터 선형 보간법을 통해 새로운 샘플을 생성하는 데이터 증강 기법
- 간단한 동작 원리에도 좋은 일반화 성능을 보장하여 다양한 딥러닝 연구 분야에서 사용되고 있는 기법

Mix-up formulation

$$\tilde{x} = \lambda x_i + (1-\lambda)x_j, \quad \text{where } x_i, x_j \text{ are raw input vectors}$$
$$\tilde{y} = \lambda y_i + (1-\lambda)y_j, \quad \text{where } y_i, y_j \text{ are one-hot label encodings}$$

$\lambda \sim Beta(\alpha, \alpha)$





Mix-up

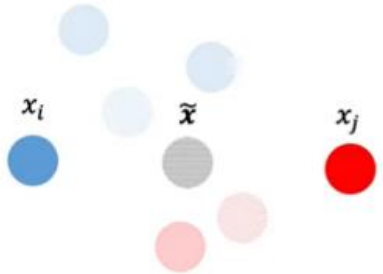
❖ Mix-up 알고리즘 동작 원리

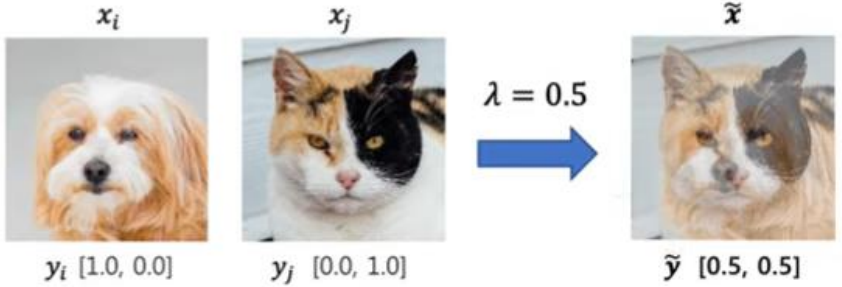
- 두 데이터 샘플로부터 선형 보간법을 통해 새로운 샘플을 생성하는 데이터 증강 기법
- 간단한 동작 원리에도 좋은 일반화 성능을 보장하여 다양한 딥러닝 연구 분야에서 사용되고 있는 기법

Mix-up formulation

$$\tilde{x} = \lambda x_i + (1-\lambda)x_j, \quad \text{where } x_i, x_j \text{ are raw input vectors}$$
$$\tilde{y} = \lambda y_i + (1-\lambda)y_j, \quad \text{where } y_i, y_j \text{ are one-hot label encodings}$$

$\lambda \sim Beta(\alpha, \alpha)$





1. MetaMix

- 즉, Hidden Layer 를 이용해서 Manifold 상의 feature vector 까지 이용한다는 것을 의미한다.
  - 어떤 model 이 L 개의 레이어로 이루어져 있을 때 아래를 만족하는 l 이 있다고 하자

$$l \in \mathcal{C} = \{0, 1, \cdots, L-1\}$$

  - 이때 Mix-up 에 기반하여 Linear interpolation 을 수행한다. 즉, Support set 에 있는 sample 과 Query set 에 있는 sample 을 합치는 것을 의미한다

$$\mathbf{X}_{i,l}^{mix} = \boldsymbol{\lambda} f_{\phi_i^l}(\mathbf{X}_i^s) + (\mathbf{I} - \boldsymbol{\lambda}) f_{\phi_i^l}(\mathbf{X}_i^q),$$
$$\mathbf{Y}_i^{mix} = \boldsymbol{\lambda} \mathbf{Y}_i^s + (\mathbf{I} - \boldsymbol{\lambda}) \mathbf{Y}_i^q,$$

  - 일반적인 Mix-up 과 마찬가지로 Beta distribution 을 따른다

$$\boldsymbol{\lambda} = \text{diag}(\{\lambda_j\}_{j=1}^{K^q}) \text{ and each coefficient } \lambda_j \sim \text{Beta}(\alpha, \beta)$$

1. MetaMix

- Sample 의 수의 경우, Spt set 과 Qry set 의 sample 수가 같다고 하였으나
  - Ks < Kq 의 경우, 각각의 query set 에 대해서, support set 에서 랜덤하게 데이터를 가져옴
  - Ks > Kq 의 경우, 각각의 support set 에 대해서, query set 에서 랜덤하게 데이터를 가져옴

- 그리하여, outer-loop 를 재정의 할 수 있다
  - Before

$$\theta_0^* := \min_{\theta_0} \frac{1}{n_T} \sum_{i=1}^{n_T} \left[\mathcal{L}(f_{\phi_i}(\mathbf{X}_i^q), \mathbf{Y}_i^q) \right],$$

  - After

$$\theta_0^* := \min_{\theta_0} \frac{1}{n_T} \sum_{i=1}^{n_T} \mathbb{E}_{\boldsymbol{\lambda} \sim \mathrm{Beta}(\alpha,\beta)} \mathbb{E}_{l \sim \mathcal{C}} [\mathcal{L}(f_{\phi_i^{L-l}}(\mathbf{X}_{i,l}^{mix}), \mathbf{Y}_i^{mix})],$$

2. MetaMix enhanced with Channel Shuffle

- Classification task 에서 CF 에 의해 더 잘 할 수 있다고 되어있다

- 어떠한 neural net 의 layer 단에서 어떠한 class 의 subset 을 다른 class 의 subset 과 바꾸어주는 역할이라고 생각하면 편함

- 특정 layer l 에 p 개의 channel 이 있다고 가정하면 아래와 같이 나타낼 수 있다

$$f_{\phi_i^l}(\mathbf{x}_{i,j}^{s(q)}) = [f_{\phi_i^l}^{(1)}(\mathbf{x}_{i,j}^{s(q)}); \ldots; f_{\phi_i^l}^{(p)}(\mathbf{x}_{i,j}^{s(q)})]$$

- 특정 class 끼리의 subset 을 아래와 같이 나타낼 수 있다

$$(\mathbf{X}_{i;c}^{s(q)}, \mathbf{Y}_{i;c}^{s(q)}), (\mathbf{X}_{i;c'}^{s(q)}, \mathbf{Y}_{i;c'}^{s(q)})$$

2. MetaMix enhanced with Channel Shuffle

- 베르누이 분포를 따르는 RV 에 의해 얼마만큼 셔플 시킬지를 정해주고 아래의 폼에 맞추어서 재정의 해주도록 한다

- 아래의 식이 Channel Shuffling 이다

$$\mathbf{X}_{i;c}^{s(q),cf} = \mathbf{R}_{c,c'} f_{\phi_i^l}(\mathbf{X}_{i;c}^{s(q)}) + (\mathbf{I} - \mathbf{R}_{c,c'}) f_{\phi_i^l}(\mathbf{X}_{i;c'}^{s(q)}),$$
$$\mathbf{Y}_{i;c}^{s(q),cf} = \mathbf{Y}_{i;c}^{s(q)}.$$

- Outer Loop 에서는 아래와 같이 쓸 수 있다 여기서 mmcf는 MetaMix + Channel shFfling 을 나타낸다

$$\mathbf{X}_{i,l}^{mmcf} = \boldsymbol{\lambda} \mathbf{X}_i^{s,cf} + (\mathbf{I} - \boldsymbol{\lambda}) \mathbf{X}_i^{q,cf},$$
$$\mathbf{Y}_i^{mmcf} = \boldsymbol{\lambda} \mathbf{Y}_i^{s,cf} + (\mathbf{I} - \boldsymbol{\lambda}) \mathbf{Y}_i^{q,cf},$$

Sudo Code (MetaMix)

Algorithm 1 Meta-training Process of MAML-MetaMix

Require: Task distribution $p(\mathcal{T})$; Learning rate μ, η; Beta distribution parameters α, β; MetaMix candidate layer set $\mathcal{C}$

Randomly initialize parameter θ_0
while not converge **do**
  Sample a batch of tasks $\{\mathcal{T}_i\}_{i=1}^{n}$
  for all $\mathcal{T}_i$ **do**
    Sample support set $\mathcal{D}_i^s = \{(\mathbf{x}_{i,j}^s, \mathbf{y}_{i,j}^s)\}_{j=1}^{K^s}$ and query set $\mathcal{D}_i^q = \{(\mathbf{x}_{i,j}^q, \mathbf{y}_{i,j}^q)\}_{j=1}^{K^q}$ from $\mathcal{T}_i$
    Compute the task-specific parameter ϕ_i via the inner-loop gradient descent, i.e., $\phi_i = \theta_0 - \mu\nabla_{\theta_0}\mathcal{L}(f_{\theta_0}(\mathbf{X}_i^s), \mathbf{Y}_i^s)$
    Sample MetaMix parameter $\boldsymbol{\lambda} \sim \mathrm{Beta}(\alpha, \beta)$ and mixed layer l from $\mathcal{C}$
    Forward both support and query sets and mixed them at layer l as: $\mathbf{X}_{i,l}^{mix} = \boldsymbol{\lambda} f_{\phi_i^l}(\mathbf{X}_i^s) + (\mathbf{I} - \boldsymbol{\lambda}) f_{\phi_i^l}(\mathbf{X}_i^q)$, $\mathbf{Y}_i^{mix} = \boldsymbol{\lambda}\mathbf{Y}_i^s + (\mathbf{I} - \boldsymbol{\lambda})\mathbf{Y}_i^q$
    Continual forward $\mathbf{X}_{i,l}^{mix}$ to the rest of layers and compute the loss as $\mathcal{L}(f_{\phi_i^{L-l}}(\mathbf{X}_{i,l}^{mix}), \mathbf{Y}_i^{mix})$
  end for
  Update $\theta_0 \leftarrow \theta_0 - \eta\frac{1}{n}\sum_{i=1}^{n}\mathbb{E}_{\boldsymbol{\lambda}\sim\mathrm{Beta}(\alpha,\beta)}\mathbb{E}_{l\sim\mathcal{C}}[\mathcal{L}(f_{\phi_i^{L-l}}(\mathbf{X}_{i,l}^{mix}), \mathbf{Y}_i^{mix})]$
end while

2. MetaMix enhanced with Channel Shuffle

Algorithm 1 Meta-training Process of MAML-MMCF

Require: Task distribution $p(\mathcal{T})$; Learning rate μ, η; Beta distribution parameters α, β; MetaMix candidate layer set $\mathcal{C}$

Randomly initialize parameter θ_0
while not converge **do**
  Sample a batch of tasks $\{\mathcal{T}_i\}_{i=1}^{n}$
  for all $\mathcal{T}_i$ **do**
    Sample support set $\mathcal{D}_i^s = \{(\mathbf{x}_{i,j}^s, \mathbf{y}_{i,j}^s)\}_{j=1}^{K^s}$ and query set $\mathcal{D}_i^q = \{(\mathbf{x}_{i,j}^q, \mathbf{y}_{i,j}^q)\}_{j=1}^{K^q}$ from $\mathcal{T}_i$
    Sample a mixed layer l from $\mathcal{C}$
    Sample Channel Shuffle parameter $\mathbf{R}_{c,c'}$ for each pair of classes c and c'
    Perform Channel Shuffle on the support set as (use a pair of classes as an example) via Eqn. (6) in the original paper: $\mathbf{X}_{i,c}^{s,cf} = \mathbf{R}_{c,c'} f_{\phi_i^l}(\mathbf{X}_{i,c}^s) + (\mathbf{I} - \mathbf{R}_{c,c'}) f_{\phi_i^l}(\mathbf{X}_{i,c'}^s)$, $\mathbf{Y}_{i,c}^{s,cf} = \mathbf{Y}_{i,c}^s$.
    Compute the task-specific parameter ϕ_i via the inner-loop gradient descent, i.e., $\phi_i = \theta_0 - \mu \nabla_{\theta_0} \mathcal{L}(f_{\theta_0}(\mathbf{X}_i^{s,cf}), \mathbf{Y}_i^{s,cf})$
    Perform Channel Shuffle on the query set via Eqn. (6) in the original paper: $\mathbf{X}_{i,c}^{q,cf} = \mathbf{R}_{c,c'} f_{\phi_i^l}(\mathbf{X}_{i,c}^q) + (\mathbf{I} - \mathbf{R}_{c,c'}) f_{\phi_i^l}(\mathbf{X}_{i,c'}^q)$, $\mathbf{Y}_{i,c}^{q,cf} = \mathbf{Y}_{i,c}^q$.
    Sample MetaMix parameter $\boldsymbol{\lambda} \sim \mathrm{Beta}(\alpha, \beta)$
    Forward both support and query sets and mixed them at layer l as: $\mathbf{X}_{i,l}^{mmcf} = \boldsymbol{\lambda} f_{\phi_i^l}(\mathbf{X}_i^{s,cf}) + (\mathbf{I} - \boldsymbol{\lambda}) f_{\phi_i^l}(\mathbf{X}_i^{q,cf})$, $\mathbf{Y}_i^{mmcf} = \boldsymbol{\lambda} \mathbf{Y}_i^{s,cf} + (\mathbf{I} - \boldsymbol{\lambda}) \mathbf{Y}_i^{q,cf}$
    Continual forward $\mathbf{X}_{i,l}^{mix}$ to the rest of layers and compute the loss as $\mathcal{L}(f_{\phi_i^{L-l}}(\mathbf{X}_{i,l}^{mmcf}), \mathbf{Y}_i^{mmcf})$
  end for
  Update $\theta_0 \leftarrow \theta_0 - \frac{1}{n} \sum_{i=1}^{n} \mathbb{E}_{\boldsymbol{\lambda} \sim \mathrm{Beta}(\alpha,\beta)} \mathbb{E}_{l \sim \mathcal{C}} [\mathcal{L}(f_{\phi_i^{L-l}}(\mathbf{X}_{i,l}^{mmcf}), \mathbf{Y}_i^{mmcf})]$
end while

MT

5 Discussion with Related Works

Discussion with Related Works

- Euclidean metric, Cosine metric 과 같이 메트릭을 이용해 분류하는 작업
  - Classification 에만 국한되어 있을 뿐, 다른 task (Regession?) 같은 거에는 힘들다
  - 본 연구는 Gradient-based meta learning 알고리즘을 사용

- Support set 으로 훈련하여, Query set 의 performance 를 높여주는 것이 목표
  - 하지만 **overfitting 이라는 high-risk** 가 존재한다
  - 또한 meta-test set 에 generalization 이 부족하다

- Overfitting 을 막기 위한 common 테크닉에는 Regularizations 이 있음
  - Weight decay
  - Dropout
  - Incorporating noise

Discussion with Related Works

- Meta learning 의 경우, parameter의 개수를 줄이거나 adapted noise 를 줄이는 방법이 있음, 하지만 이와 같이 inner-loop 의 overfitting 을 해결하는 법은 좋은 방법이지만, 제한적이다

- 좀 더 최근에는 2가지 방법이 제안됨
  - MR-MAML
    - initialization 의 search space 를 규제함
  - TAML
    - task 간에 비슷하게 행동하도록 초기화를 시행

- 이러한 방법 대신에, spt set 과 qry set 에 노이즈를 추가하는 방법이 있으나, 단순 추가하는 방법으로는 Second criterion 을 만족하지 못한다

6 Experiments

6.1 Drug Activity Prediction

- A real-world application of drug activity prediction
- 4,276 target tasks → Consists of few drug compounds with tested activities against the target protein
- Evaluate the square of Pearson coefficient R2 between the predicted yˆ_q i and the ground-truth y_q of all query samples for each i-th task
- Report the mean and median R2 values over all meta-testing assays as well as the number of assays with R2 > 0.3 which is deemed as an indicator of reliability in pharmacology

Performance

- Notice that only updating the final layer in the inner-loop achieves the best performance, which is equivalent to ANIL

- MetaMix consistently improves the performance despite of the backbone meta-learning algorithms

- Demonstrates that
  - MetaMix is compatible with existing meta-learning algorithms;
  - MetaMix is capable of improving the meta-generalization ability.

- Investigate the influence of different hyperparameter settings (e.g., α in Beta(α, α)), and demonstrate the robustness of MetaMix under

Analysis of Overfitting

- 아래의 figure는 meta-train, meta-test 의 Performance 를 나타냄

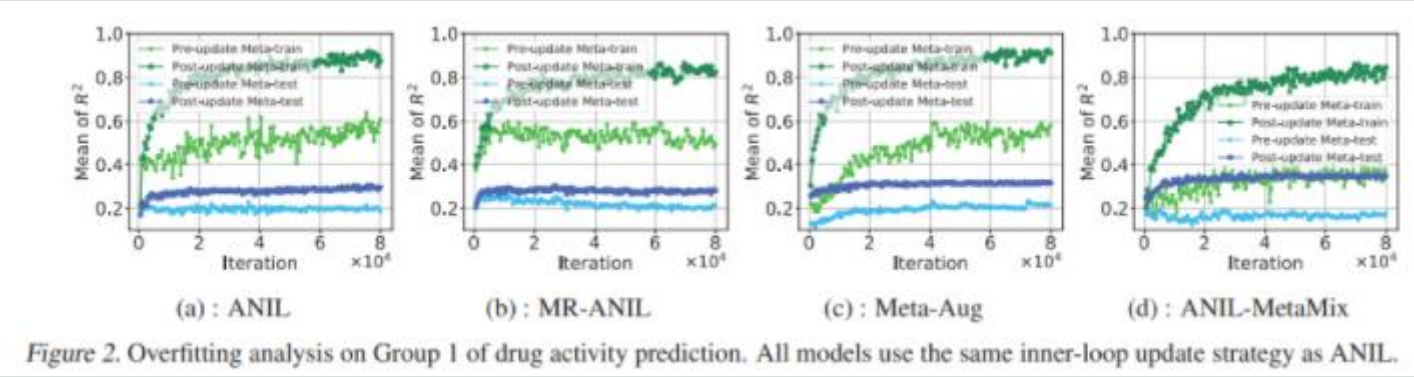


Figure 2. Overfitting analysis on Group 1 of drug activity prediction. All models use the same inner-loop update strategy as ANIL.

- **MetaMix significantly increases the performance gap** between pre-update (θ_{0}) and post-update (ϕ_{i}),
  - **MetaMix improves the dependence of target prediction on support sets**, and therefore alleviates memorization overfitting

Analysis of Overfitting

- 아래의 figure는 meta-train, meta-test 의 Performance 를 나타냄

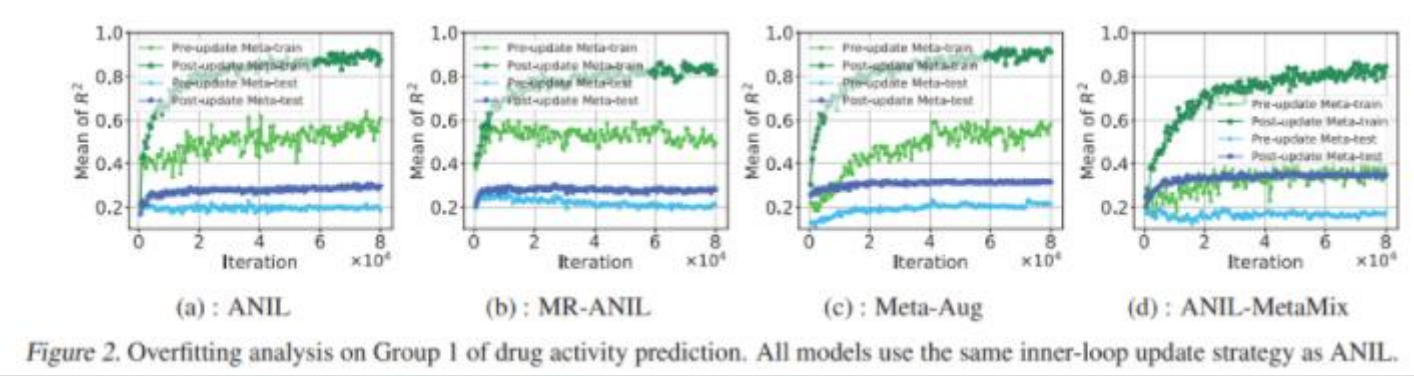


Figure 2. Overfitting analysis on Group 1 of drug activity prediction. All models use the same inner-loop update strategy as ANIL.

- Compared to Meta-Aug and MR-ANIL, the worse pre-update meta-training performance but **better post update meta-testing performance of MetaMix** demonstrates its superiority to mitigate the learner overfitting

  → Meta-testing 시 성능이 많이 올라서 1등찍긴했는데...차이가 미비하다

Effect of Data Mixture Strategy in MetaMix

- 어디까지 개선되는지를 확인하기 위해, 5가지의 mix 된 전략을 사용하였다

Table 2. Effect of mixture strategies on drug activity prediction. All strategies are applied on ANIL++.

Strategies	Group 1			Group 2			Group 3			Group 4		
	Mean	Med.	>0.3	Mean	Med.	>0.3	Mean	Med.	>0.3	Mean	Med.	>0.3
$\mathcal{D}^q$	0.367	0.299	50	0.315	0.252	43	0.335	0.289	48	0.362	0.324	51
Set Shuffle	0.371	0.352	55	0.293	0.224	42	0.339	0.297	50	0.360	0.300	50
Mixup($\mathcal{D}^s$, $\mathcal{D}^s$)	0.224	0.164	33	0.210	0.164	31	0.214	0.154	29	0.191	0.141	22
Mixup($\mathcal{D}^q$, $\mathcal{D}^q$)	0.388	0.354	55	0.322	0.264	46	0.341	0.306	50	0.358	0.325	53
$\mathcal{D}^{cob} = \mathcal{D}^s \oplus \mathcal{D}^q$	0.376	0.324	52	0.301	0.242	44	0.333	0.329	51	0.336	0.281	48
MetaMix	**0.413**	**0.393**	**59**	**0.337**	**0.301**	**51**	**0.381**	**0.362**	**55**	**0.380**	**0.348**	**55**

- D^{cob} 는 Concat 을 나타내는데, 이것의 성능이 의미하는 바는 단순히 합하기만해서는 노답이라는 것을 의미한다

Analysis of Criteria

Table 3. Criteria analysis on Group 1 of drug activity prediction. All models use ANIL as the backbone meta-learning algorithm.

Aug. Method	C1	C2	H(Y|X)↑	Mean R^2
Mix-All				0.292
Mixup($\mathcal{D}^q$, $\mathcal{D}^q$)		√	√	0.322
Meta-Aug	√		√	0.317
ANIL-MetaMix	√	√	√	**0.347**

- Further analyze **augmentation methods on drug data (Group 1)** with respect to the two criteria (C1, C2) we propose and the CE-increasing criterion H(Y|X)↑ proposed by Meta-Aug
- C1, C2 2개의 criteria 가 모두 중요함을 알 수 있음

6.2 Pose Prediction

- **Regression dataset** created from Pascal 3D data, where a 128×128 grey-scale image is used as input and the orientation relative to a fixed pose labels each image
- 50 and 15 objects are randomly selected for meta-training and meta-testing

Result

- More data → more effectiveness than meta-regulaizer

- MAML-MetaMix의 성능이 Meta-Aug보다 우수하다는 것은 θ_{0} 를 학습하기 위한 추가적인 지식을 가져오는 것의 효과를 추가로 검증한 것과 마찬가지

Table 4. Performance (MSE ± 95% confidence interval) of pose prediction.

Model	10-shot	15-shot
Weight Decay	2.772 ± 0.259	2.307 ± 0.226
CAVIA	3.021 ± 0.248	2.397 ± 0.191
Meta-dropout	3.236 ± 0.257	2.425 ± 0.209
Meta-Aug	2.553 ± 0.265	2.152 ± 0.227
MR-MAML	2.907 ± 0.255	2.276 ± 0.169
TAML	2.785 ± 0.261	2.196 ± 0.163
ANIL	6.746 ± 0.416	6.513 ± 0.384
MAML	3.098 ± 0.242	2.413 ± 0.177
MetaSGD	2.803 ± 0.239	2.331 ± 0.182
T-Net	2.835 ± 0.189	2.609 ± 0.213
ANIL-MetaMix	6.354 ± 0.393	6.112 ± 0.381
MAML-MetaMix	2.438 ± 0.196	2.003 ± 0.147
MetaSGD-MetaMix	$\mathbf{2.390 \pm 0.191}$	$\mathbf{1.952 \pm 0.154}$
T-Net-MetaMix	2.563 ± 0.201	2.418 ± 0.182

7 Conclusion

Conclusion

- To address poorly generalizing to meta-testing tasks, propose two novel data augmentation strategies
  - MetaMix
  - Channel Shuffle

  That is actively involve more data in the outer-loop optimization process

- MetaMix → linearly interpolates the features and labels of support and target sets
- Channel Shuffle → randomly replaces a subset of channels with the corresponding ones from another class

Conclusion

- Theoretically demonstrate that all strategies can improve the meta-generalization capability
- The state-of-the art results on different real-world datasets demonstrate the effectiveness and compatibility of the proposed methods.

Thank you