Journal of KIIT. Vol. 23, No. 12, pp. 79-88, Dec. 31, 2025. pISSN 1598-8619, eISSN 2093-7571 79
http://dx.doi.org/10.14801/jkiit.2025.23.12.79

신경망 학습을 위한 동적 re-Batch

이석원*, 강태원**

Dynamic re-Batch for Neural Network Training

Seokwon Lee*, Taewon Kang**

요 약

인공 신경망의 학습 효율과 성능을 개선하기 위한 기존의 특징 추출 및 샘플링 방법은 도메인별 휴리스틱이나 복잡한 알고리즘에 의존하여 범용성이 제한되는 한계가 있다. 본 논문에서는 학습이 쉽거나 어려운 샘플을 손실값을 지표로 일정 비율 선정하여, 각 학습 에포크마다 학습 데이터세트에 동적으로 재분배하는 동적 re-Batch 전략을 제안한다. 실험 결과, 최소 손실 동적 re-Batch는 안정적인 경향을 보이며 이진 분류 및 일부 회귀 문제에서 성능을 개선하였다. 반면, 최대 손실 동적 re-Batch는 대다수의 실험에서 학습 불안정성과 성능 저하를 일으킨 것으로 나타났다. 이는 학습하기 어려운 샘플이 노이즈나 이상치일 가능성을 시사한다. 이 전략의 효과는 데이터세트에 따라 최적의 재분배 비율을 찾는 것이 중요함을 확인하였다.

Abstract

Existing feature extraction and sampling methods for improving the learning efficiency and performance of artificial neural networks rely on domain-specific heuristics or complex algorithms, which limits their generalizability. We propose a dynamic re-Batch strategy that selects a certain proportion of samples with easy or difficult learning based on loss values, dynamically redistributing them in the training dataset for each training epoch. Results showed that the minimum loss dynamic re-Batch is generally stable and improves performance in classification and some regression problems. In contrast, the maximum loss dynamic re-Batch caused learning instability and performance degradation in most cases, suggesting that difficult samples may be noise or outliers. The effectiveness of this strategy has been confirmed by the importance of finding the optimal redistribution ratio depending on the dataset.

Keywords
dynamic re-Batch, loss-based sampling, learning stability, hyperparameter tuning

* 강릉원주대학교 산업대학원 석사과정
- ORCID: https://orcid.org/0009-0005-8755-4325
** 강릉원주대학교 컴퓨터공학과 교수(교신저자)
- ORCID: https://orcid.org/0000-0003-4343-5517

· Received: Sep. 11, 2025, Revised: Oct. 15, 2025, Accepted: Oct. 18, 2025
· Corresponding Author: Taewon Kang
Dept. of Computer Science & Engineering, Gangneung-Wonju National University. Korea
Tel.: +82-33-760-8666, Email: twkang@gwnu.ac.kr

Ⅰ. 서 론

인공 신경망 기반의 딥러닝 기술은 의료, 시각 인식, 자연어 처리 등 다양한 분야에서 우수한 성과를 나타내고 있으나, 실제 문제의 복잡성과 데이터의 다양성으로 인해 학습 효율성과 모델 성능을 향상하기 위한 연구는 여전히 중요한 과제로 제기된다. 특히, 데이터로부터 유의미한 특징을 효과적으로 추출하고 학습하는 전략은 모델의 수렴 속도와 일반화 성능을 좌우하는 핵심 요소로 간주된다[1].

그러나 기존 방법론들은 몇 가지 한계를 지닌다. 예컨데, 모델 학습에서 샘플이 학습에 얼마나 어려운지 나타내는 지표를 정의하기 위해 도메인에 특화된 복잡한 휴리스틱이 요구되거나, 복잡한 샘플링 과정, 손실 가중치 부여, 또는 균형 제어 알고리즘을 사용한다. 이러한 복잡성은 다양한 데이터세트와 문제에 범용적인 적용이 제한될 수 있다.

이러한 한계를 극복하고자 동적 re-Batch 전략을 제안한다. 이 전략은 학습하는 샘플의 손실값이라는 단일하고 범용적인 지표만을 사용하여 샘플의 학습 난이도를 정의한다. 이는 별도의 복잡한 사전 분석 없이 다양한 데이터세트에 적용 가능하다. 또한, 특정 손실 구간에 속하는 샘플들을 선정하여 전체 학습 데이터에 일정한 간격으로 재분배하는 단순한 절차를 사용하였다.

나아가, 최소 손실(Min-loss) 동적 re-Batch와 최대 손실(Max-loss) 동적 re-Batch라는 두 가지 전략을 동시에 탐구한다. 이러한 접근을 통해 각 전략이 모델의 수렴 안정성, 그리고 최종 성능에 미치는 영향을 이진 분류와 회귀 문제에서 종합적으로 비교·분석함으로써, 동적 re-Batch 전략의 근본적인 효과와 실용적 시사점을 제시하고자 한다.

Ⅱ. 관련 연구

본 연구의 동적 re-Batch 전략은 기존의 다양한 학습 데이터 샘플링 및 가중치 부여 방법론들과 유사한 맥락을 가진다. 커리큘럼 학습(Curriculum learning)은 모델에 대해 쉬운 샘플부터 점진적으로 어려운 샘플을 제시함으로써 학습 효율을 높이는 전략이다[2]. 하드 예시 마이닝(Hard example mining)은 모델이 학습하기 어려워하는 예시에 집중하여 학습 효율 향상을 목표로 하는 전략이다[3]. TSW (Top-q Sample Weighting)은 손실값이 큰 샘플에 가중치를 부여하여 모델이 중요한 정보에 더 집중하도록 유도하는 방법론이다[4]. Breed(Balance ratio and enhancE density)은 손실값을 기반으로 샘플링 밀도를 조절하고, 균일 샘플링과의 비율을 고려해 학습 안정성을 확보하는 알고리즘이다[5].

동적 re-Batch와 기존 방법론과의 차이점을 비교하면 표 1과 같다. 동적 re-Batch 전략은 학습하는 샘플의 손실값이라는 단일하고 범용적인 지표만을 사용하여 샘플이 학습하기 쉬운지 어려운지 정의한다. 이는 별도의 사전 분석이나 복잡한 설계 없이 다양한 데이터세트와 문제에 용이하게 적용될 수 있다는 장점을 제공한다.

표 1. 기존 방법론과 차이점 요약
Table 1. Summary of differences with existing methodologies

Methodology	Key concept	Differences
Curriculum learning	Learn from easy sample	Use a single metric called loss value without complex heuristics
Hard example mining	Focus on difficult samples	Using a simple redistribution mechanism without a separate complex sampling process
TSW	Assigning weights according to losses	Redistributing samples at regular intervals without weighting
Breed	Loss-based sampling density adjustment	Redistribution only with the redistribution ratio without balance control value

또한, 샘플 재분배 메커니즘이 단순하다. 복잡한 샘플링 과정, 손실 가중치 부여 혹은 균형 제어 알고리즘을 사용하는 것과는 달리, re-Batch 전략은 특정 손실 구간에 속하는 샘플들을 선정하여, 전체 학습 데이터에 일정한 간격으로 재분배하는 단순한 절차를 사용한다.

Ⅲ. 동적 re-Batch

동적 re-Batch 전략은 테이블 형태의 데이터를

1D CNN 모델에 적용하기 위한 전처리 과정, 1D CNN, 동적 re-Batch를 이용한 배치 학습 이 세 가지 요소로 구성된다.

PyTorch 프레임워크를 이용하여 구현하였으며, 데이터 로더는 데이터세트에서 배치를 구성하여 모델에 전달하는 역할을 수행한다[6]. 일반적으로는 shuffle=True와 같은 옵션을 통해 데이터의 순서를 지정하지만, 이 순서를 동적으로 변경하는 복잡한 절차는 샘플러가 담당한다. 샘플러는 데이터세트에서 샘플의 인덱스를 이용하여 샘플을 추출한 뒤 데이터 로더에 전달하는 역할을 한다[7]. IntervalInsertionSampler는 샘플러를 상속 받아 구현하였다. 먼저, 학습 데이터세트에서 재분배 인덱스 집합에 포함되지 않은 나머지 샘플의 인덱스들을 추출한다. 이후, 이 나머지 샘플의 인덱스들에 재분배 샘플들을 일정한 간격으로 삽입하여 최종적인 훈련 순서를 완성한다. 이 간격(Interval)은 식 (1)에 따라 나머지 인덱스 수(Nremaining)를 배치 크기(B)로 나눈 값의 내림으로 계산된다.

$$Interval = \lfloor \frac{N_{remaining}}{B} \rfloor \quad (1)$$

재분배 과정을 그림 1에 제시하였다.

표 2는 IntervalInsertionSampler의 동적 재분배 기능이 어떻게 작용하는지를 보여준다. 학습 루프 내에서 각 에포크가 종료된 후, 다음 에포크를 위한 데이터 재분배 단계가 실행된다.

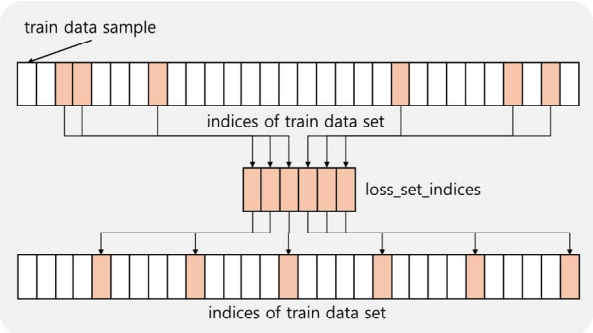

그림 1. 동적 re-Batch의 샘플 재분배 과정 예시 이미지
Fig. 1. Example image of the process of repositioning data samples for re-Batch strategy

$$B = \lfloor \frac{N_{train} \times r}{1+r} \rfloor \quad (2)$$

배치 크기는 학습 데이터 수(N_{train})와 재분배 비율(r)에 따라 결정되며, 식 (2)와 같이 정의된다. 이는 재분배 대상이 될 샘플의 수에 해당한다.

각 샘플의 손실값은 BCEWithLogitsLoss[8], MSELoss[9] 손실 함수의 reduction='none' 매개변수를 이용하여 PyTorch의 GPU 행렬 연산의 이점을 유지하였다.

표 2. Base trainer의 손실 기반 동적 re-Batch 실행 절차 의사 코드
Table 2. Pseudocode for the loss-based dynamic re-Batch procedure of the base trainer

```
Input:  Training dataset 'train_dataset', batch training data loader 'train_loader', sample data loader for batch
        'sample_loader', 'min-loss' or 'max-loss' strategy,
Output: The model has beed trained. 'model'
1:  for epoch from 1 to 'num_epochs':
3:      // Batch training stage
4:      for each batch in 'train_loader':
5:          Performing model training and error backpropagation in batch.
6:          // Loss-based sample selection stage.
7:          for each 'sample' in 'sample_loader':
8:              all_losses_indices = Calculate the loss and index of the sample.
9:          if strategy is 'min-loss'
10:             'sorted_indices = Sort the index in ascending order based on the sample loss.
11:         else 'max-loss':
12:             'sorted_indices' = Sort the index in descending order based on the sample loss.
13:         'loss_set_indices' = Select only the top index set from the batch size in 'sorted_indices'
14:     Reconstruct 'train_loader' and 'sample_loader' using 'intervalInsertionSampler' and 'loss_set_indices' for
15: the next epoch.
16:     Evaluate the model using a validation dataset.
    return 'model'
```

재분배 샘플 선별은 최소 손실 re-Batch 전략의 경우 오름차순 정렬, 최대 손실 re-Batch 전략의 경우 내림 차순 정렬하여 정렬 결과에서 상위에 위치한 샘플 인덱스를 재분배 비율만큼 선정하였다.

선정된 재분배 샘플 인덱스들은 IntervalInsertionSampler로 전달되어, 다음 에포크 훈련을 위한 데이터 로더를 재구성하는 데 사용된다. 이로써 모델은 다음 에포크부터 재분배된 샘플들을 포함하는 새로운 데이터 순서로 학습을 수행하게 된다.

각 데이터의 특성은 평균 0, 분산 1로 표준화하였다. 1D CNN은 배치 크기, 채널 수, 시퀀스 길이의 3차원 입력을 요구한다[10]. 이에 따라 1차원 데이터에 새로운 차원을 추가하였다. 이 연산을 통해 '(특성 수)' 형태의 1차원 데이터가 '(1, 특성 수)' 형태의 2차원 텐서로 변환된다. 1D CNN은 3차원 입력에서 채널을 1로, 시퀀스 길이를 특성 수로 간주한다. 최종적으로 이러한 2차원 텐서들을 묶어 '(배치 크기, 1, 특성 수)' 형태의 3차원 텐서를 모델에 입력으로 제공하였다.

1D CNN 모델은 두 개의 블록으로 구성하였다. 각 블록은 중첩층, ReLU 활성화 함수, 최대 풀링층으로 이루어져 있다.

예측 헤드는 특성 추출부의 출력을 1차원 벡터로 평탄화한 후, 2개의 완전 연결층을 통과하도록 설계하였다. 첫 번째 층은 특성 추출부의 특성 벡터 길이를 입력으로 받고, 입력 특성의 차원을 2배로 확장하여 두 번째 층에 전달한다. 이 과정에서 ReLU 활성화 함수를 적용하였다. 이진 분류의 경우 평탄화 벡터를 입력받아 두 개의 선형층을 거쳐 이진 분류를 위한 단일 로짓값을 출력하였다. 회귀의 경우 선형층을 사용하지만, 회귀 문제에 대해 단일 연속값을 예측하였다.

Ⅳ. 성능 평가

4.1 데이터세트

손실 기반 동적 re-Batch 전략의 효과를 정량적으로 분석하기 위해 다양한 특징을 가진 총 여섯 개의 데이터세트를 사용하여 성능을 평가하였다. 각 데이터세트는 이진 분류 및 회귀 문제로 구분되며, 각 실험 시나리오에 대해 반복 실험을 수행하여 결과의 신뢰성을 확보하였다. 실험에 사용된 데이터세트는 모두 UCI 머신러닝 저장소 및 캐글 등 공개 출처에서 수집하였다. 모든 데이터세트는 각 특성의 표준화 과정을 거쳤다. 데이터는 훈련, 검증, 테스트 세트로 각각 6:2:2의 비율로 분할하였다.

이진 분류 문제로는 PID(Pima Indians Diabetes)[11], CCFD(Credit Card Fraud Detection)[12], PDC(Parkinson's Disease Classification)[13] 를 사용하였으며, CCFD는 데이터 불균형이 심각하므로 정상 거래 샘플의 약 0.8%만을 임의로 샘플링하였다.

표 3. 데이터세트에 따른 신경망 구조
Table 3. Neural network structure according to the dataset

Data set	Block	Layer	Parameter	Value
PID, CCFD	1	conv1d	out_channels	8
			kernel_size	3
			stride	1
		maxPooling	kernel_size	1
			stride	1
	2	conv1d	in_channels	8
			out_channels	16
			kernel_size	3
			stride	1
		maxPooling	kernel_size	3
			stride	2
PDC	1	conv1d	out_channels	8
			kernel_size	8
			stride	1
		maxPooling	kernel_size	2
			stride	2
	2	conv1d	in_channels	8
			out_channels	16
			kernel_size	8
			stride	1
		maxPooling	kernel_size	2
			stride	2
Diabetes, insurance	1	conv1d	out_channels	64
			kernel_size	3
			stride	1
		maxPooling	kernel_size	2
			stride	2
	2	conv1d	in_channels	64
			out_channels	128
			kernel_size	3
			stride	1
		maxPooling	kernel_size	2
			stride	2
CHP	1	conv1d	out_channels	32
			kernel_size	3
			stride	1
		maxPooling	kernel_size	2
			stride	2
	2	conv1d	in_channels	32
			out_channels	64
			kernel_size	3
			stride	1
		maxPooling	kernel_size	2
			stride	2

회귀 문제로는 Diabetes[14], CHP(California Housing Price)[15], Insurance[16] 를 사용하였으며, CHP는 전체 20,640개의 샘플 중 10%를 임의로 샘플링하여 사용하였다.

표 3은 데이터세트에 따른 신경망 구조를 제시한다. 항상 첫 번째 중첩층은 입력 채널을 1로 설정하고, 시퀀스 길이는 해당 데이터세트의 특성 수에 맞추어 구성하였다.

4.2 학습 및 평가 과정

모든 실험은 Adam Optimizer를 사용하였으며, 학습률은 0.001로 고정하였다. 손실 함수는 이진 분류 문제에 BCEWithLogitsLoss를, 회귀 문제에 MSELoss를 적용하였다. 각 데이터세트의 특성을 고려하여 충분한 수렴을 위해 에포크 수를 PID 300회, CCFD 50회, PDC 30회, Diabetes 200회, CHP 200회, Insurance 100회로 설정하였다. 재분배 비율은 0.03, 0.08, 0.15, 0.3을 각각 적용하였다. 각 실험 시나리오당 10회 반복 실행 후 그 평균 결과를 분석에 활용하였다. 또한, 모든 실험은 동일한 임의 시드(Seed)를 42로 고정하여 수행하였다.

모델의 성능은 이진 분류의 경우 정확도, 정밀도, 재현율, F1-점수를 사용하였다. 회귀의 경우 평균 제곱근 오차, 평균 절대 오차, 결정계수 오차 그리고 정규화된 평균 제곱근 오차를 사용하였다. Diabetes의 경우 표준편차 기반, CHP와 Insurance는 평균 기반 정규화된 평균 제곱근 오차를 사용하였다.

각 손실 곡선(훈련 손실, 검증 손실)에 대해 선형 회귀를 적용하여 기울기와 절편을 계산하였으며, 이를 통해 손실 감소 추세의 안정성을 정량화하였다.

또한, re-Batch 전략(β)과 re-Batch를 사용하지 않은 전략(α)의 손실 곡선 간 면적 비교(Loss area)를 통해 수렴 안정성을 분석하였다. 손실 면적은 식 (3)과 같이 계산한다.

$$loss\ area = \alpha - \beta \tag{3}$$

V. 실험 결과

5.1 이진 분류 문제

표 4, 표 5, 표 6에 각각 PID, CCFD, PDC 데이터세트에 대한 실험 결과를 제시하였다.

표 4. PID 성능 비교(좌), 손실 영역 비교(우)
Table 4. Comparison of performance(Left), loss areas(right) for PID

Type	Ratio	A	P	R	F1	Type	Ratio	Loss type	Loss area
no re-Batch	0.03	0.70	0.56	0.59	0.58	min-loss re-Batch	0.03	train	1.26
	0.08	0.70	0.57	0.56	0.56			val	-2.75
	0.15	0.71	0.57	0.6	0.58		0.08	train	4.37
	0.3	0.71	0.57	0.6	0.58			val	-20.19
min-loss re-Batch	0.03	0.69	0.55	0.55	0.55		0.15	train	7.21
	0.08	0.71	0.57	0.61	0.59			val	-59.88
	0.15	0.72	0.57	0.64	0.60		0.3	train	16.31
	0.3	0.71	0.58	0.59	0.58			val	-60.43
max-loss re-Batch	0.03	0.69	0.55	0.55	0.55	max-loss re-Batch	0.03	train	1.4
								val	-74.07
	0.08	0.71	0.57	0.61	0.59		0.08	train	1.27
								val	-97.31
	0.15	0.72	0.57	0.64	0.60		0.15	train	-22.57
								val	-43.48
	0.3	0.71	0.58	0.59	0.58		0.3	train	-36.89
								val	-30.38
TSW	0.03	0.69	0.54	0.54	0.54	TSW	0.03	train	-27.3
								val	-434.2
	0.08	0.70	0.56	0.57	0.57		0.08	train	-94.5
								val	-124.9
	0.15	0.69	0.55	0.53	0.54		0.15	train	-148.3
								val	36.8
	0.3	0.72	0.58	0.60	0.59		0.3	train	-157.3
								val	14.1

* A : Accuracy, P : Precision, R : Recall, F1 : F1-Score

표 5. CCFD 성능 비교(좌), 손실 영역 비교(우)
Table 5. Comparison of performance(Left), loss areas(Right) for CCFD

Type	Ratio	A	P	R	F1	Type	Ratio	Loss type	Loss area
no re-Batch	0.03	0.95	0.88	0.83	0.85	min-loss re-Batch	0.03	train	-0.17
	0.08	0.95	0.91	0.83	0.87			val	-0.22
	0.15	0.96	0.93	0.82	0.87		0.08	train	-0.12
	0.3	0.95	0.93	0.81	0.87			val	-0.08
min-loss re-Batch	0.03	0.95	0.94	0.81	0.87		0.15	train	0.19
	0.08	0.96	0.94	0.82	0.88			val	-0.13
	0.15	0.96	0.93	0.83	0.88		0.3	train	0.45
	0.3	0.96	0.94	0.82	0.88			val	-0.04
max-loss re-Batch	0.03	0.95	0.91	0.83	0.87	max-loss re-Batch	0.03	train	-0.78
								val	-1.11
	0.08	0.95	0.93	0.82	0.87		0.08	train	-3.83
								val	-1.23
	0.15	0.95	0.92	0.83	0.87		0.15	train	-7.23
								val	-2.15
	0.3	0.95	0.92	0.81	0.86		0.3	train	-12.91
								val	-3.22
TSW	0.03	0.95	0.92	0.83	0.87	TSW	0.03	train	-23
								val	-4.8
	0.08	0.95	0.90	0.83	0.86		0.08	train	-35
								val	-11
	0.15	0.95	0.89	0.83	0.86		0.15	train	-38
								val	-15
	0.3	0.95	0.89	0.84	0.86		0.3	train	-38
								val	-17

* A : Accuracy, P : Precision, R : Recall, F1 : F1-Score

표 6. PDC 성능 비교 (좌), 손실 영역 비교(우)
Table 6. Comparison of PDC performance(Left), loss areas(Right) for PDC

Type	Ratio	A	P	R	F1	Type	Ratio	Loss type	Loss area
no re-Batch	0.03	0.89	0.89	0.94	0.92	min-loss re-Batch	0.03	train	0.2
	0.08	0.88	0.90	0.92	0.91			val	-0.1
	0.15	0.88	0.89	0.93	0.91		0.08	train	0.51
	0.3	0.87	0.88	0.93	0.9			val	-0.37
min-loss re-Batch	0.03	0.89	0.90	0.93	0.92		0.15	train	0.4
	0.08	0.88	0.90	0.93	0.91			val	-0.02
	0.15	0.87	0.89	0.93	0.91		0.3	train	0.62
	0.3	0.86	0.88	0.92	0.9			val	0.03
max-loss re-Batch	0.03	0.9	0.9	0.94	0.92	max-loss re-Batch	0.03	train	0.03
								val	1.05
	0.08	0.88	0.89	0.93	0.91		0.08	train	-0.23
								val	-0.38
	0.15	0.88	0.89	0.94	0.91		0.15	train	1.25
								val	-1.04
	0.3	0.86	0.87	0.93	0.9		0.3	train	-2.07
								val	-0.82
TSW	0.03	0.9	0.91	0.94	0.92	TSW	0.03	train	0.13
								val	-4.99
	0.08	0.89	0.89	0.95	0.92		0.08	train	-2.99
								val	-2.29
	0.15	0.88	0.89	0.93	0.91		0.15	train	-7.38
								val	0.3
	0.3	0.88	0.89	0.93	0.91		0.3	train	-9.6
								val	-0.4

* A : Accuracy, P : Precision, R : Recall, F1 : F1-Score

최소 손실 동적 re-Batch 전략은 전반적으로 안정적인 성능 개선을 보인 것으로 나타났다. 특히 CCFD 데이터세트의 경우, 재분배 비율에 관계없이 일관되게 정확도와 F1-점수가 향상되는 결과를 보였다. 이는 모델에 이미 학습한 쉬운 샘플을 반복적으로 노출함으로써 학습의 안정성을 높이고 일반화

성능을 개선하였음을 시사한다. 이 전략은 모델이 데이터의 핵심적인 패턴을 놓치지 않고 견고하게 학습하도록 유도한 것으로 해석된다.

반면, 최대 손실 동적 re-Batch 전략은 모든 분류 데이터세트에서 학습 불안정성과 성능 저하를 초래한 것으로 나타났다. 이는 손실값이 큰 학습하기 어려운 샘플만으로 배치를 구성하는 것이 오히려 모델 학습에 방해되는 노이즈나 이상치를 반복 학습하게 된 결과로 추정된다.

TSW 전략은 PID, CCF 데이터세트의 일부 배치 크기에서 성능이 더 우수하다. PDC 데이터세트에서는 모든 배치 크기에서 성능이 더 우수한 반면, 각 데이터세트 전반적으로 학습 및 검증 손실은 동적 re-Batch 전략에 비해 더 크게 나타났다.

5.2 회귀 문제

표 7, 표 8, 표 9에 각각 Diabetes, CHP, Insurance 데이터세트에 대한 실험 결과를 제시하였다.

Insurance 데이터세트에서는 재분배 비율 0.15, 0.3에서 RMSE/MAE가 크게 감소하며 훈련 효율성과 일반화 능력 모두 향상되었다.

표 7. Diabetes 성능 비교(좌), 손실 영역 비교(우)
Table 7. Comparison of performance(Left), loss areas(Right) for Diabetes

re-Batch type	Ratio	A	B	C	D	Type	Ratio	Loss type	Loss area
no re-Batch	0.03	74	55	0.04	0.97	min-loss re-Batch	0.03	train	-3.7E4
	0.08	69	52	0.17	0.91			val	-4.4E4
	0.15	63	48	0.29	0.83		0.08	train	-1.8E4
	0.3	60	48	0.37	0.78			val	1.4E4
min-loss re-Batch	0.03	82	65	-0.19	1.09		0.15	train	3.9E5
	0.08	67	51	0.21	0.88			val	7.5E3
	0.15	64	49	0.27	0.85		0.3	train	7.6E4
	0.3	60	47	0.36	0.79			val	2.0E4
max-loss re-Batch	0.03	78	61	-0.05	1.02	max-loss re-Batch	0.03	train	-4.8E4
								val	-2.6E4
	0.08	73	56	0.07	0.96		0.08	train	-9.2E4
								val	-3.9E4
	0.15	65	51	0.26	0.85		0.15	train	-2.9E5
								val	-3.8E4
	0.3	60	48	0.37	0.79		0.3	train	-5.3E5
								val	-4.3E4

* A : RMSE, B : MAE, C : R2E, D : NRMSE

표 8. CHP 성능 비교(좌), 손실 영역 비교(우)
Table 8. Comparison of performance(Left), loss areas(Right) for CHP

re-Batch type	Ratio	A	B	C	D	Type	Ratio	Loss type	Loss area
no re-Batch	0.03	0.71	0.46	0.63	0.35	min-loss re-Batch	0.03	train	1.45
	0.08	0.79	0.45	0.54	0.38			val	0.4
	0.15	0.7	0.44	0.64	0.34		0.08	train	4.32
	0.3	0.65	0.42	0.69	0.32			val	-2.68
min-loss re-Batch	0.03	0.72	0.48	0.56	0.38		0.15	train	1.71
	0.08	0.68	0.46	0.63	0.35			val	-0.59
	0.15	0.7	0.44	0.67	0.33		0.3	train	10.03
	0.3	0.5	0.43	0.65	0.34			val	6.21
max-loss re-Batch	0.03	0.79	0.48	0.55	0.38	max-loss re-Batch	0.03	train	-2.92
								val	-2.43
	0.08	0.72	0.46	0.62	0.35		0.08	train	-13.42
								val	-5.04
	0.15	0.72	0.46	0.62	0.35		0.15	train	-46.86
								val	-11.6
	0.3	0.68	0.44	0.67	0.33		0.3	train	-87.15
								val	-15.49

* A : RMSE, B : MAE, C : R2E, D : NRMSE

표 9. Insurance 성능 비교
Table 9. Comparison of performance(Left), loss areas(Right) for Insurance

Type	Ratio	A	B	C	D	Type	Ratio	Loss type	Loss area
no re-Batch	0.03	4.9E3	2.8E3	0.84	0.34	min-loss re-Batch	0.03	train	1.0E8
	0.08	5.2E3	3.2E3	0.82	0.36			val	5.4E6
	0.15	6.7E3	4.8E3	0.71	0.47		0.08	train	1.7E8
	0.3	7.9E3	6.1E3	0.59	0.56			val	1.4E8
min-loss re-Batch	0.03	5.0E3	3.4E3	0.83	0.35		0.15	train	4.5E8
	0.08	5.1E3	3.2E3	0.83	0.36			val	4.4E8
	0.15	6.3E3	4.5E3	0.74	0.45		0.3	train	8.4E8
	0.3	7.3E3	5.5E3	0.65	0.51			val	7.5E8
max-loss re-Batch	0.03	5.4E3	3.8E3	0.81	0.38	max-loss re-Batch	0.03	train	-1.7E9
								val	-4.0E8
	0.08	5.8E3	4.2E3	0.78	0.41		0.08	train	-4.9E9
								val	-3.8E8
	0.15	7.5E3	5.8E3	0.63	0.52		0.15	train	-1.1E10
								val	-1.0E9
	0.3	8.2E3	6.4E3	0.56	0.57		0.3	train	-2.1E10
								val	-1.4E9

* A : RMSE, B : MAE, C : R2E, D : NRMSE

그러나 Diabetes와 CHP 데이터세트에서는 일부 비율에서만 효과적이었으며, 그 외에는 오히려 성능 저하가 발생하였다. 이는 쉬운 샘플을 반복 학습하는 과정이 회귀 문제의 복잡한 분포를 파악하는 방해 요인으로 작용할 수 있음을 시사하는 결과이다.

최대 손실 re-Batch 전략은 회귀 문제에서도 심각한 성능 저하를 초래하였다. 특히 Insurance 데이터세트의 경우, 모든 재분배 비율에서 RMSE/MAE가 상승하여 학습이 거의 불가능한 상태에 도달하였다. 이는 최대 손실 샘플이 대부분 이상치이거나 데이터 분포의 극단에 위치한 샘플로 구성되어, 모델의 수렴을 저해하는 요인으로 작용한 것으로 추정된다.

VI. 결론 및 향후 과제

최소 손실 동적 re-Batch 전략은 최대 손실 동적 re-Batch에 비해 안정적이고 효과적인 학습 방법으로 확인되었다. 이 전략은 모델이 이미 학습한 쉬운 샘플을 반복적으로 제시하여, 학습의 안정성을 높이고 데이터의 핵심적인 패턴에 대한 견고한 이해를 유도한다. 특히, 회귀 문제에서는 Insurance 데이터세트에서 가장 큰 성능 개선 효과를 보였다. 그러나 일부 데이터세트에서는 재분배 비율이 지나치게 낮을 경우 효과가 제한적이거나, 특정 조건에서는 과적합이 관찰되기도 했다.

최대 손실 동적 re-Batch 전략은 대부분의 실험에서 학습 불안정성 및 성능 저하를 일으키는 것으로 나타났다. 이 전략의 근간이 되는 학습하기 어려운 샘플이 실제로 모델이 아직 학습하지 못한 정보성 높은 샘플이 아닌, 단순한 노이즈나 이상치일 가능성이 높음을 시사하는 결과를 보였다. 이러한 학습 데이터 샘플에 모델의 자원을 집중하는 것은 학습을 방해하고 모델의 예측을 왜곡할 수 있다. 특히, 회귀 문제에서는 분류 문제보다 부정적인 영향이 더 심각하게 나타났는데, 이는 회귀의 손실 함수가 이상치에 더 민감하게 반응하여 모델이 소수의 극단적인 샘플에 과도하게 편향된 결과로 해석된다.

본 연구의 결과는 동적 re-Batch 전략이 효과적일 수 있으나, 그 효과는 데이터세트의 특성과 문제 유형에 따라 크게 달라진다는 중요한 시사점을 제공한다. 최적의 재분배 비율은 데이터세트의 크기, 복잡성 정도에 따라 달라진다. 작은 비율의 샘플을 반복 학습하는 것이 더 효과적일 수도 있고, 때로는 더 많은 샘플을 재분배하는 것이 나을 수도 있다.

다음의 연구를 통해 동적 re-Batch 전략의 실용성을 높일 것이다. 첫째, 학습 단계에 따라 재분배 비율을 점진적으로 조정하여, 초기 안정성 확보와 후반부 다양성 확대라는 두 목표를 동시에 달성하는

방안을 모색한다. 둘째, 최소 손실 샘플의 안정성과 최대 손실 샘플의 정보성을 균형 있게 활용하는 새로운 하이브리드 전략을 개발한다. 셋째, 향후 각 태스크에 맞는 최적의 비율을 효율적으로 탐색하고 결정하는 체계적인 방법론을 찾는다.

References

[1] I. H. Saker, "Deep Learning: A Comprehensive Overview on Techniques, Taxonomy, Applications and Research Directions", SN Computer science, Vol. 2, No. 420, Aug. 2021. https://doi.org/10.1007/s42979-021-00815-1.

[2] P. Soviany, R. T. Ionescu, P. Rota, and N. Sebe, "Curriculum Learning: A Survey", International Journal of Computer Vision, Vol. 130, No. 19, pp. 1526-1565, Jun. 2022. https://doi.org/10.1007/s11263-022-01611-x.

[3] A. Shrivastava, A. Gupta, and R.Girshick, "Training Region-based Object Detectors with Online Hard Example Mining", Proc. IEEE Conf. on Computer Vision and Pattern Recognition(CVPR), Las Vegas, USA, pp. 761-769, Jun. 2016. https://doi.org/10.48550/arXiv.1604.03540.

[4] Y. Wang and W. Wo, "Ordered sample weighting: Improving neural network training for complex image classification", Discrete and Continuous Dynamical Systems - S, Early Access, Dec. 2024. https://doi.org/doi:10.3934/dcdss.2024210.

[5] S. Dymchenko and B. Raffin, "Loss-driven sampling within hard-to-learn areas for simulation-based neural network training", Proc. Proc MLPS Workshop at NeurIPS 2023 – 37th Conf. on Neural Information Processing Systems, New Orleans, USA, pp. 1-5, Dec. 2023. https://hal.science/hal-04305233v1.

[6] PyTorch torch.utils.data.DataLoader, https://docs.pytorch.org/docs/stable/data.html#torch.utils.data.DataLoader. [accessed: Aug. 27, 2025]

[7] PyTorch torch.utils.data.Sampler, https://docs.pytorch.org/docs/stable/data.html#torch.utils.data.Sampler. [accessed: Aug. 28, 2025]

[8] PyTorch torch.nn.BCEWithLogitsLoss, https://docs.pytorch.org/docs/stable/generated/torch.nn.BCEWithLogitsLoss.html. [accessed: Oct. 11. 2025]

[9] PyTorch torch.nn.MSELoss, https://docs.pytorch.org/docs/stable/generated/torch.nn.MSELoss.html. [accessed: Oct. 11. 2025]

[10] PyTorch torch.nn.Conv1d, https://docs.pytorch.org/docs/stable/generated/torch.nn.Conv1d.html. [accessed: Aug. 28. 2025]

[11] J. W. Smith, J. E. Everhart, W. C. Dickson, W. C. Knowler, and R. S Johannes, "Using the ADAP Learning Algorithm to Forecast the Onset of Diabetes Meliitus", Proc. Symposium on Computer Applications and Medical Care, pp. 261-265, Nov. 1988.

[12] A. D. Pozzolo, O. Caelen, R. A. Johnson, and G. Botempi, "Calibrating Probability with Undersampling for Unbalanced Classification", Proc. IEEE Symposium Series on Camputational Intelligence, Cape Town, South Africa, pp. 1-8, Dec. 2015. http://doi.org/10.1109/SSCI.2015.33.

[13] UCI Machine Learning Repository Parkinsons Dataset, https://archive.ics.uci.edu/dataset/174/parkinsons. [accessed: Aug. 28, 2025]

[14] B. Efron, T Hastie, I Johnstone, and R Tibshirani "Least angle regression", Institute of Mathematical Statistics in The Annals of Statistics, Vol. 32, No. 2, pp. 407-451, Apr. 2004. https://doi.org/10.1214/009053604000000067.

[15] R. K. Pace and R. Berry, "Sparse spatial autoregressions", Statistics & Probability Letters, Vol. 33, No. 3, pp. 291-297, May 1997. https://doi.org/10.1016/S0167-7152(96)00140-X.

[16] Kaggle Medical Cost Personal Datasets, https://www.kaggle.com/datasets/mirichoi0218/insurance. [accessed: Aug. 28, 2025]

저자소개

이 석 원 (Seokwon Lee)

2020년 2월 : 강릉원주대학교 컴퓨터공학과(학사)
2020년 2월 ~ 2021년 7월 : (주)케이에스씨엔씨 전임
2021년 8월 ~ 2025년 4월 : (주)디엑스솔루션즈 파트장
2025년 3월 ~ 현재 : 강릉원주대학교 컴퓨터공학과 산업대학원 석사과정
관심분야 : 복잡계, 인공생명, 인공지능

강 태 원 (Taewon Kang)

1985년 2월 : 연세대학교, 서울 수학과(이학사)
1988년 2월 : 고려대학교, 서울 전산과학과(이학사)
1991년 2월 : 고려대학교, 서울 수학과(이학석사)
1996년 8월 : 고려대학교, 서울 컴퓨터학과(이학박사)
1997년 3월 ~ 현재 : 강릉원주대학교 컴퓨터공학과 교수
관심분야 : 복잡계, 인공생명, 인공지능, 소프트컴퓨팅