ミスラベルデータの忘却による学習済みモデルの汎化性能の向上手法の提案

杉浦　一瑳[†]　　岡村　真吾[††]　　山下　恭佑[†]　　矢内　直人[†]

† 大阪大学 〒 565-0871 大阪府吹田市山田丘 1-1
†† 奈良工業高等専門学校 〒 639-1080 奈良県大和郡山市矢田町 22 番地
E-mail: †{i-sugiura,yamashita,yanai}@ist.osaka-u.ac.jp, ††okamura@info.nara-k.ac.jp

あらまし　機械学習モデルはミスラベルデータを学習することで汎化性能が低下する．ミスラベルデータに対する既存の対策手法は，学習前あるいは学習中の処理が前提となっていた．本稿では，ミスラベルデータの忘却による学習済みモデルの汎化性能の向上手法を提案する．提案手法は大まかには，まず学習済みモデルから計算できる異常度を通じてミスラベルデータを検知したのち，検知したデータをモデルから忘却 (Machine Unlearning) させることで，汎化性能を向上させる．提案手法について，ベンチマークとして MNIST データセットを用いた分類モデルによる実験を行った．手法の有効性を詳細に評価するために，まず検知と忘却に分けてそれぞれ実験したところ，ミスラベルデータの混入割合が少ないときに検知が有効であること，また，ミスラベルデータを忘却することで汎化性能が向上することを確認した．しかし，提案手法に相当する検知と忘却を繋げた実験では，ミスラベルデータの混入割合に応じて検知の性能が低下することに起因し，汎化性能の向上が妨げられることがわかった．

キーワード　ミスラベルデータ，Machine Unlearning，検知，忘却

1　はじめに

機械学習は訓練データの増大とともに，性能の向上を遂げてきた．しかし，クラウドソーシングを活用して収集された大量のデータにはラベルの誤ったデータ (ミスラベルデータ) を含んでしまうことや [1–3]，ラベルを意図的に変えたデータを訓練データの一部に加える訓練データ汚染攻撃 [4, 5] の存在などの理由から，正確にラベル付けされた質の良い訓練データの増大は容易ではない．ミスラベルデータはモデルの汎化性能を低下させることや [6, 7]，プライバシー漏洩を助長させることが指摘されており [8–10]，ミスラベルデータへの対策は重要である．

上述した背景に対し，訓練データセットに含まれるミスラベルデータの影響を抑える手法が研究されてきた [11–15]．しかし，これらはモデルの学習における前処理，あるいは学習中の処理として用いられる手法であり，既存の学習済みモデルに対して事後的に適用することが難しい．近年は Hugging Face [16] や GitHub [17] などのプラットフォームを通じて学習済みモデルを公開することが増えているため，既存の学習済みモデルに対しても適用可能な手法が望ましい．

本稿では，ミスラベルデータを含んだデータセットを用いて学習した学習済みのモデルに対し，事後的にミスラベルデータを検知し，そのミスラベルデータを除いて再学習するよりも効率的に忘却させる手法を提案する．提案手法の概要を図 1.1 に示す．提案手法は大まかに検知，忘却の 2 段階で構成される．具体的には，まず訓練データの各サンプルに対して損失や影響関数 [18] 等の異常度を計算することでミスラベルデータを検知する．その後，検知したデータの影響を取り除くようにモデルのパラメータを更新することで，ミスラベルデータを忘却する．忘却には，学習済みモデルから訓練データの一部を忘れさせる Machine Unlearning という分野において提案されている近似的な忘却手法を用いる [18, 19]．本提案手法により，既存の学習済みモデルに対して，ミスラベルデータの影響を取り除き，汎化性能を向上させる．

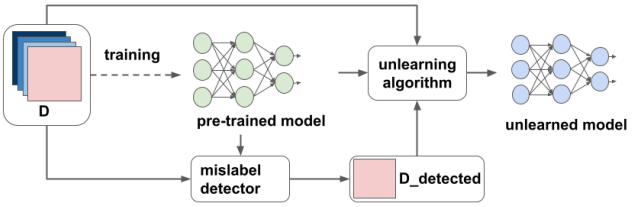

図 1.1: 提案手法の概要

提案手法の有効性を確認するため，分類モデルを対象にベンチマークである MNIST データセットを用いて実験を行った．はじめに，提案手法の有効性を詳細に分析するため，提案手法の段階ごとに実験を行った．結果として，まずミスラベルデータの混入割合が少ないときにミスラベルデータの検知が有効であること，また，パラメータを更新することでミスラベルデータの影響を抑えられることをそれぞれ確認した．しかし，提案手法に相当するミスラベルデータの検知とパラメータ更新を繋げた実験では，提案手法の効果が確認できなかった．この原因として，混入割合が増加するにつれて，検知の段階で，ミスラベルデータの検知の性能が下がっていることを確認した．

本稿の貢献を要約すると，以下のとおりである．

- ミスラベルデータを効率的に忘却する手法を提案した.
- 分類モデルを対象に実験を行ったところ, ミスラベルデータの混入割合が少ないときにミスラベルデータの検知が有効であることを確認した.
- 近似的な忘却手法を用いてミスラベルデータを忘却すると汎化性能が向上することを確認した.
- 提案手法に相当するミスラベルデータの検知とパラメータ更新を繋げた実験では, 汎化性能の向上が確認できなかった.
- 主な原因として, 混入割合が増加するにつれ, 元となるミスラベルデータの検知性能が下がっていることを確認した.

2 関 連 研 究

本節ではミスラベルデータを伴う学習手法, および, 忘却手法としての Machine Unlearning についてそれぞれ述べる. 次に, 本研究に最も近い関連研究として, 機械学習モデルの復元手法について述べる.

2.1 ミスラベルデータを伴う学習手法

ミスラベルデータを伴う学習手法は, 大量のデータを必要とするがラベルにノイズが含まれる可能性があるという状況で, 性能の高いモデルを得るために研究されてきた [20, 21]. とくに前処理にあたるミスラベルデータを検知する手法 [13–15] や, 中処理にあたるミスラベルデータの影響を軽減する手法 [11, 12] が提案されている.

ミスラベルデータを前処理として検知する手法として, モデルの学習中の予測値 $f(x;\theta)$ の時間発展がミスラベルデータと正しくラベル付けされたデータで異なることを利用して, ミスラベルデータを検知する手法が提案されている [14, 15].

ミスラベルデータの影響を中処理として軽減する手法としては, ラベルの遷移確率を扱う手法 [11] や, ラベル誤りにロバストな損失関数を設計する手法 [12] が存在する. しかし, これらの手法は学習方法をあらかじめ変更することを前提としているため, 本稿とは問題設定が異なっている.

2.2 Machine Unlearning

Machine Unlearning [22] は, 機械学習モデルから訓練データの一部の影響を取り除く手法である. 元はプライバシーの文脈で使われており, 対象のデータがモデルの訓練データに含まれているか否かを推論するメンバーシップ推論 [23] や, モデルから学習データを復元するデータ抽出 [24] など, 機械学習モデルに対するプライバシー攻撃の危険性が注目されるにつれて重要性が増している. 近年では学習データの汚染攻撃の対策としても注目されている [25]. Machine Unlearning を実現する素朴な方法は, 忘れさせたいデータを取り除いた上で再学習を行うことである. しかし, この方法は時間的および計算的コストが大きいため, 現実的ではない. そのため, 効率的に Machine Unlearning を実現する方法が研究されている [18, 19, 26]. 本稿では, 学習手法を変更する必要のない忘却手法として, 影響関数を用いた手法 [18], および, ニュートン法を用いた手法 [19] を用いる. 以降ではそれらについて説明する.

2.2.1 影響関数 (Influence Function) を用いた手法

影響関数は Interpretable ML でも用いられる手法で, データセット中のデータ z のモデルへの影響を近似的に図る手法である [18].

影響関数では, 訓練データセットを $\mathcal{D} = \{z_i = (x_i, y_i)\}_{i=1}^n$ としたときのデータ $z \in \mathcal{D}$ の損失を ϵ で重み付けして増やしたときのパラメータの変化を考える. すなわち,

$$\hat{\theta}_{\epsilon,z} := \arg\min_{\theta\in\Theta} \frac{1}{n}\sum_{i=1}^{n} L(z_i,\theta) + \epsilon L(z,\theta) \tag{2.1}$$

としたとき, $\hat{\theta}_{\epsilon,z} - \hat{\theta}$ を考える. このとき, 以下のように近似的にパラメータの ϵ に関する $\hat{\theta}_{\epsilon,z} - \hat{\theta}$ の微分を計算できる.

$$\mathcal{I}_{up,params}(z) := \left.\frac{d\hat{\theta}_{\epsilon,z}}{d\epsilon}\right|_{\epsilon=0} = -H_{\hat{\theta}}^{-1}\nabla_\theta L(z,\hat{\theta}) \tag{2.2}$$

ただし, $H_{\hat{\theta}} := \frac{1}{n}\sum_{i=1}^{n}\nabla_\theta^2 L(z_i,\hat{\theta})$. データ z を削除することは式 (2.1) において $\epsilon = -\frac{1}{n}$ とすることと同じであるため, データ z を取り除いた上で再学習した結果得られる最適解は, $\hat{\theta}^{\setminus z}$ $\hat{\theta}^{\setminus z} \approx \hat{\theta} - \frac{1}{n}\mathcal{I}_{up,params}(z)$ で計算できる. ただし, 上記の方法は近似であり, 導出における仮定に沿わない深層学習モデルでは誤差が大きくなる. また, 逆行列 $H_{\hat{\theta}}^{-1}$ の計算コストが大きいという問題がある.

2.2.2 ニュートン法を用いた手法 (SSSE)

ニュートン法を用いた手法 (SSSE) は, 忘却対象のデータを $\mathcal{D}_{forget}, |\mathcal{D}_{forget}| = m$ とした時, 以下のようにパラメータを更新する手法である [19].

$$\hat{\theta}^{\setminus\mathcal{D}_{forget}} = \hat{\theta} + \frac{1}{n-m}H_{\hat{\theta},\mathcal{D}\setminus\mathcal{D}_{forget}}^{-1}\nabla_\theta L(\mathcal{D}_{forget},\hat{\theta}) \tag{2.3}$$

ただし, $H_{\hat{\theta},\mathcal{D}\setminus\mathcal{D}_{forget}}^{-1} := \frac{1}{n-m}\sum_{(x_i,y_i)\in\mathcal{D}\setminus\mathcal{D}_{forget}}\nabla_\theta^2 L(z_i,\hat{\theta})$. この方法も $H_{\hat{\theta},\mathcal{D}\setminus\mathcal{D}_{forget}}^{-1}$ の計算コストが大きいが, 計算を工夫することで計算コストを削減することができ, 再学習による忘却に比べて効率的に忘却することができると考えられている.

2.3 ノイズを含むデータで学習した学習済みモデルの修正手法

ノイズを含むデータで学習した学習済みモデルの修正手法は, 訓練データ汚染攻撃 [27] に対する防御手法として広く研究されてきた. 訓練データ汚染攻撃とは, 攻撃者が訓練データセットにラベルを意図的に変更したり, トリガーを含めたりしてモデルの挙動を変える攻撃である. 訓練データ汚染攻撃に対する防御手法として, 汚染データの検知およびモデルの修正が提案されている [28–30].

KARMA [28] は元のデータセットと誤分類結果のレポートを入力とし, 誤分類の原因となったデータを特定し, 忘れさせるフレームワークである. KARMA と提案手法との比較を表 2.1

表 2.1: 既存研究との比較

	モデル	目的	忘却対象
KARMA [28]	SVM 等	誤推論結果の修正	誤推論の原因のデータ
提案手法	DNN	汎化性能の向上	ミスラベルデータ

に示す. 本稿の研究では誤分類結果ではなくモデルの汎化性能を向上することに焦点が置かれている点, 想定するモデルを深層学習モデルとしている点で, 問題設定が KARMA と異なっている. すなわち, 本稿の方がより一般化された問題とみなせる.

また, トリガーを含む汚染データに対する検知, 修正手法が提案されている [29,30]. これらの研究において, 修正方法として汚染データを取り除いた上で再学習する手法が用いられている. これに対し, 本稿ではモデルの修正として近似的な Machine Unlearning を取り入れた点に新規性がある.

3 問題設定

本章では, 提案手法が必要となる問題設定について説明する.

3.1 問題の定式化

以下に, 問題設定を定式化する. 以降では, クラス分類問題の教師あり学習とする. まず入力データの集合 $\mathcal{X}$, ラベルの集合 $\mathcal{Y}$, 機械学習モデルの訓練データセットを $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n \subset \mathcal{X} \times \mathcal{Y}$ とする. ここで機械学習モデルは $f : \mathcal{X} \times \Theta \to \mathcal{Y}$ なる写像であり, データセット $\mathcal{D}$ のうち, ミスラベルデータからなるデータセットを $\mathcal{D}_{mislabeled} \subset \mathcal{D}$, そうでないサンプル（便宜上, クリーンデータと呼称）からなるデータセットを $\mathcal{D}_{clean} \subset \mathcal{D}$ とし, このとき $\mathcal{D} = \mathcal{D}_{mislabeled} \cup \mathcal{D}_{clean}$ となる. 各サンプル $z_i = (x_i, y_i)$ の損失を $L(z_i, \theta)$ とし, $L(\mathcal{D}, \theta) := \frac{1}{n}\sum_i^n L(z_i, \theta)$ とする. データセット $\mathcal{D}$ を用いてパラメータ θ を学習した結果得られたパラメータの値を $\hat{\theta}$ とし, 学習済みモデルを $f(\cdot; \hat{\theta})$ とする.

本研究では, データセット $\mathcal{D}$, 学習済みモデル $f(\cdot; \hat{\theta})$, ミスラベルデータの混入割合 (noise ratio) $r = \frac{|\mathcal{D}_{mislabeled}|}{|\mathcal{D}|}$ から, $\mathcal{D}_{mislabeled}$ を可能な限り忘却することで, モデルの汎化性能を向上させることを目指す. ここでいう忘却とは, Machine Unlearning の研究で考えられている, $\mathcal{D}$ を用いて学習した学習済みモデルのパラメータを, 忘却対象のデータ $\mathcal{D}_{forget}$ に対して, $\mathcal{D} \setminus \mathcal{D}_{forget}$ を用いて学習した結果得られるパラメータに近づけるという意味とする [31].

なお, データセット $\mathcal{D}$ を知っているという状況は, 学習済みモデルの提供者や, 提供時に用いたデータセットがオープンであり公開されていた場合に起きる. また, 混入割合 r を知っているという状況は, 混入割合を正確に推定するのは困難であるが, データセットの数が大きい場合, データセットの一部をとってきて混入割合を調査し, 全体の混入割合を推定することで可能となる.

3.2 ユースケース

本問題設定のユースケースは以下が考えられる. (1): 意図の有無に関わらずミスラベルデータが訓練データセットに含まれる. (2): モデル提供者が訓練データセットを用いて学習済みモデル, 使用した訓練データセットを Hugging Face や GitHub に公開する. (なお, Hugging Face はデータセット [1]やモデル [2] 等を公開することのできるプラットフォームである. 例として, LLM 勉強会 [3]は, Hugging Face 上で学習済みモデルを公開しており, その際に使用した訓練データとデータ収集方法を再現可能な形で公開している [4].) (3): モデル提供者, または利用者が本提案手法を用いてモデルの汎化性能を向上する.

4 提案手法

本節では提案手法について述べる. まずは全体像を述べたのち, 手法を詳細に述べる.

4.1 全体像

提案手法は大まかには, (1) ミスラベルデータの検知と, (2) その忘却という 2 つの段階からなる.

検知はまずモデルのパラメータに対し, 入力として与えられたサンプルとそのラベルの異常度を計算し, 異常度の大きいサンプルを忘却の対象とする. 異常度の計算は, モデルを通じて計算される微分値を用いることで, 学習済みモデルに対しても可能である. 具体的には損失 (Loss), あるいは影響関数 (Influence Function) を用いる.

忘却は, 検知の段階で忘却の対象として挙げられたサンプルの影響を取り除くよう学習済みモデルのパラメータを更新する. なお, パラメータの更新には, 学習済みモデルから効率的に訓練データの一部を忘れさせる Machine Unlearning [31] という分野で提案されている, 近似的な忘却手法を用いる.

4.2 手法の詳細

提案手法の詳細として, ミスラベルデータの検知段階と忘却段階の具体的な手法をそれぞれ述べたのち, 提案手法について述べる.

4.2.1 ミスラベルデータの検知

ミスラベルデータの検知をアルゴリズム 1 に示す. このアルゴリズムでは訓練データセット $\mathcal{D}$, 学習済みモデル $f(\cdot; \hat{\theta})$, ミスラベルデータの混入割合 r を入力として, 各データ $z_i = (x_i, y_i)$ に対して, 異常度 $a(z_i, \hat{\theta})$ を計算し, 異常度の上位 $m = |D| * r$ 個のデータをミスラベルデータとみなす. 異常度の計算には, 既存研究 [14,15] をもとに考えた損失に基づく手法, あるいは, 影響関数に基づく手法 [22] をそれぞれ用いることができる.

a） 損失に基づく検知手法

ミスラベルデータはエポック数が十分に大きくなるまで, クリーンデータに比べて損失が大きいことが実験的に知られている [14,15]. そこで, ミスラベルデータがモデルに過学習されていないことを仮定し, 各データ $z_i = (x_i, y_i) \in \mathcal{D}$ に対して, 損

1：https://huggingface.co/datasets
2：https://huggingface.co/models
3：https://llm-jp.nii.ac.jp/
4：https://huggingface.co/llm-jp/llm-jp-13b-v1.0

Algorithm 1 ミスラベルデータの検知

Input: 訓練データセット $\mathcal{D}$, 学習済みモデル $f(\cdot;\hat{\theta})$, 混入割合 r
Output: ミスラベルデータとして検知された m 個のデータ $\mathcal{D}_{detected}$

1: $\mathcal{D}_{detected} \leftarrow \emptyset$
2: **for** $i = 1, 2, \ldots, n$ **do**
3: $\quad z_i = (x_i, y_i)$ の異常度 $a(z_i, \hat{\theta})$ を計算
4: **end for**
5: $\mathcal{D}_{detected} \leftarrow$ 異常度の上位 $m = |D| * r$ 個のデータ z_i を $\mathcal{D}_{detected}$ に追加

Algorithm 2 データの忘却

Input: 学習済みモデル $f(\cdot, \hat{\theta})$, $\mathcal{D}$, $\mathcal{D}_{forget}$
Output: $\hat{\theta}^{\setminus \mathcal{D}_{forget}}$

1: 忘却アルゴリズムを U とする.
2: $\hat{\theta}^{\setminus \mathcal{D}_{forget}} \leftarrow U(D, \mathcal{D}_{forget}, \hat{\theta})$

失 $L(z_i, \hat{\theta})$ を計算することで, その値を異常度とする. すなわち, $z \in \mathcal{D}$ の異常度 $a(z, \hat{\theta})$ を以下の通り定義する:

$$a(z, \hat{\theta}) := L(z, \hat{\theta}). \tag{4.1}$$

b) 影響関数に基づく検知手法

影響関数を用いた手法は, あるデータ $z_i = (x_i, y_i) \in \mathcal{D}$ をデータセットから除いたときの, モデルのパラメータ $\hat{\theta}$ の変化量を近似的に計算する手法である [18]. テストデータ $z_{test} = (x_{test}, y_{test})$[5]に対する z を取り除いた時の損失の変化量を, 以下の式で近似的に計算することができる.

$$\begin{aligned} I_{up,loss}(z_{test}, z) &:= L(z_{test}, \hat{\theta}) - L(z_{test}, \hat{\theta}^{\setminus z}) \\ &= \frac{1}{n} \nabla_\theta L(z_{test}, \hat{\theta})^T H_{\hat{\theta}}^{-1} \nabla_\theta L(z, \hat{\theta}), \end{aligned} \tag{4.2}$$

ただし,

$$H_{\hat{\theta}} = \frac{1}{n} \sum_{i=1}^{n} \nabla_\theta^2 L(z_i, \hat{\theta}) \tag{4.3}$$

である. そこで, 損失の変化量を異常度として捉えるため, 異常度 $a(z, \hat{\theta})$ を以下のように定義する.

$$a(z, \hat{\theta}) := \nabla_\theta L(z_{test}, \hat{\theta})^T H_{\hat{\theta}}^{-1} \nabla_\theta L(z, \hat{\theta}) \tag{4.4}$$

なお, 式 (4.2) の $\frac{1}{n}$ は各サンプルに対する異常度の大小関係に影響しないため, 無視している. 次の章の実験では, テストデータ z_{test} を, z, D, $D \setminus z$ の 3 つを用いて検知性能を比較する. $I_{up,loss}(z_{test}, z)$ の z_{test} として, $z, D_{train}, D_{train} \setminus z$ とした時の値を $I_{up,loss}(z, z)$, $I_{up,loss}(\mathcal{D}_{train}, z)$, $I_{up,loss}(\mathcal{D}_{train} \setminus z, z)$ として検知性能を比較する.

4.2.2 データの忘却

データの忘却をアルゴリズム 2 に示す. このアルゴリズムでは, 忘却対象のデータセット $\mathcal{D}_{forget}$, 学習済みモデル $f(\cdot, \hat{\theta})$, 訓練に用いたデータセット $\mathcal{D}$ を入力として, $\mathcal{D}_{forget}$ の影響を取り除くようにパラメータ $\hat{\theta}$ を更新する. 忘却アルゴリズムとしては, 近似的な忘却アルゴリズムである, ファインチューニング, 影響関数に基づく手法 [18], ニュートン法に基づく手法 [19] をそれぞれ用いる. なお, 実験では理想的な忘却手法である再学習を比較対象として用いる.

a) 再 学 習

再学習は, $\mathcal{D}_{forget}$ を除いたデータセット $D \setminus \mathcal{D}_{forget}$ を用いて, モデルのパラメータを初期化して初めから学習する手法である. Machine Unlearning 研究において, 理想的な忘却手法として比較対象に用いられる [31].

b) ファインチューニング

ファインチューニングは, パラメータの初期値を $\hat{\theta}$ として, $\mathcal{D}_{forget}$ を除いたデータセット $\mathcal{D} \setminus \mathcal{D}_{forget}$ を用いて, モデルのパラメータを追加学習する手法である.

c) 影響関数に基づく忘却手法

影響関数に基づく忘却手法は, 忘れさせたいデータを $z \in \mathcal{D}$ としたとき, 以下のようにパラメータを更新する [18].

$$\hat{\theta}^{\setminus \mathcal{D}_{forget}} = \hat{\theta} + \frac{1}{n} H_{\hat{\theta}}^{-1} \nabla_\theta L(z, \hat{\theta}) \tag{4.5}$$

d) ニュートン法に基づく忘却手法 (SSSE)

ニュートン法を用いた手法は, 忘れさせたいデータ $\mathcal{D}_{forget}, (|\mathcal{D}_{forget}| = m)$ とした時, 以下のようにパラメータを更新する [19].

$$\hat{\theta}^{\setminus \mathcal{D}_{forget}} = \hat{\theta} + \frac{1}{n-m} H_{\hat{\theta}, \mathcal{D} \setminus \mathcal{D}_{forget}}^{-1} \nabla_\theta L(\mathcal{D}_{forget}, \hat{\theta}). \tag{4.6}$$

4.2.3 ミスラベルデータの検知と忘却を繋げたモデル修正

提案手法は, 図 1.1 のように, ミスラベルデータの検知と忘却を繋げてモデルの汎化性能をを向上する. すなわち, ミスラベルデータの検知によって得られた $\mathcal{D}_{detected}$ を, 忘却における忘却対象 $D_{forget} = \mathcal{D}_{detected}$ として忘却する.

5 実　　験

本節では提案手法を実験により評価する. まず実験の目的を述べたのち, 設定を述べる. 最後に, 実験結果とその考察を併せて述べる.

5.1 実 験 目 的

実験の目的は提案手法の有効性を確認することにあるが, 提案手法の各段階で, それぞれ実験を行う. すなわち, 目的は以下の三点を明らかにすることにある: 1) 学習済みモデルを用いて事後的なミスラベルデータの検知は可能か; 2) ミスラベルデータに関連するパラメータを更新することで推論精度は向上するか; 3) ミスラベルデータの検知とパラメータの更新を繋げることで推論精度は向上するか. とくに上述した 3) は提案手法の全体の処理に相当することから, これが明らかにできたとき, 提案手法の有効性は確認できたとみなせる. また, 1) と 2) は提案手法の各段階に問題がないか, 明らかにする意図がある. 以降では上述した目的ごとに実験をそれぞれ行う.

5.2 実 験 設 定

本実験では, 手書き数字認識タスク用のデータセットである

5：ただし, テストデータ z_{test} は与えられていないため, z_{test} をどのように選ぶかが問題となる.

MNIST [32] を使用する．MNIST は各画像が 28 × 28 ピクセルのグレースケール画像で，0 から 9 までの 10 クラスのラベルが付与される．あらかじめ訓練データとテストデータに分割され，訓練データには 60,000 枚の画像が，テストデータには 10,000 枚の画像が含まれる．本実験では，簡単のため，0, 1 の 2 クラス分類問題として実験を行う．訓練データは各クラスのデータを 1000 個ずつ用いることにする．

ミスラベルデータを含んだデータセットの構築は，混入割合 r が与えられたとき，各ラベル y_i を以下のように変更することで行う．

$$y_i = \begin{cases} y_i & \text{with probability } 1-r, \\ 1-y_i & \text{with probability } r. \end{cases} \tag{5.1}$$

すなわち，確率 r でラベルを入れ替える．このノイズ混入方法は symmetric lable noise という名前でミスラベル研究でよく用いられている [13].

分類モデルは中間層が 2 層の Multi Layer Perceptron(MLP)，損失関数はクロスエントロピー誤差を用いる．中間層の次元数はそれぞれ 50 とし，実装は自動微分を備えた数値計算ライブラリである JAX [33]，最適化，深層学習用の JAX の派生ライブラリである Optax [34]，Flax [35] を用いた．最適化手法は SGD(確率的勾配降下法)，学習率は 0.01，バッチサイズは 32，エポック数は 20，モメンタムは 0.9 とした．

なお，本実験を再現可能なコードを GitHub にて公開している [6].

5.3 実験結果

以下に各実験の結果と考察をそれぞれ記載する．

5.3.1 実験 1: 学習済みモデルを用いて事後的なミスラベルデータの検出は有効か

学習済みモデルを用いて事後的なミスラベルデータの検出が有効であるか調べるため，混入割合 $r = \{0.01, 0.05, 0.1, 0.2, 0.3, 0.4, 0.5\}$ とし，各検知手法の性能を ROC AUC，正解率 (accuracy)，適合率 (precision) で測る．なお，今回はミスラベルとして検知する数を混入割合 r としており，この場合適合率 (precision) と再現率 (recall) が一致するため，適合率のみ示す．

混入割合に伴う各検知手法の性能を表 5.1 に示す．いずれの手法も，混入割合 r が増加するにつれて，検知性能が低下していることがわかる．また，損失を用いた手法が最も良い性能を出していることがわかる．

各サンプルの異常度の分布を可視化するため，混入割合 $r = 0.05$ の場合の学習後の各サンプルに対する損失，$I_{up,loss}(z_{test}, z)$ の z_{test} に $z, D_{train}, D_{train} \setminus z$ とした時の $I_{up,loss}(z, z)$, $I_{up,loss}(\mathcal{D}_{train}, z)$, $I_{up,loss}(\mathcal{D}_{train} \setminus z, z)$ のヒストグラムをそれぞれ図 5.1, 図 5.2, 図 5.3, 図 5.4 に示す．混入割合が小さい場合は，ミスラベルデータの異常度が大きく，ミスラベルデータとクリーンデータを分離することができることがわかる．

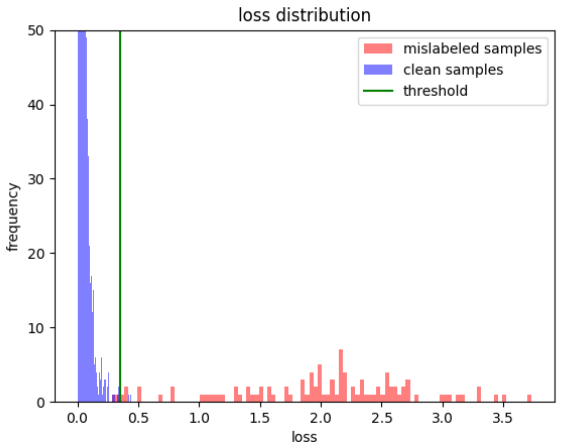

図 5.1: 混入割合 $r = 0.05$ の学習後の各サンプルに対する損失のヒストグラム．見やすさのため，y 軸の上限を 50 までとしている．

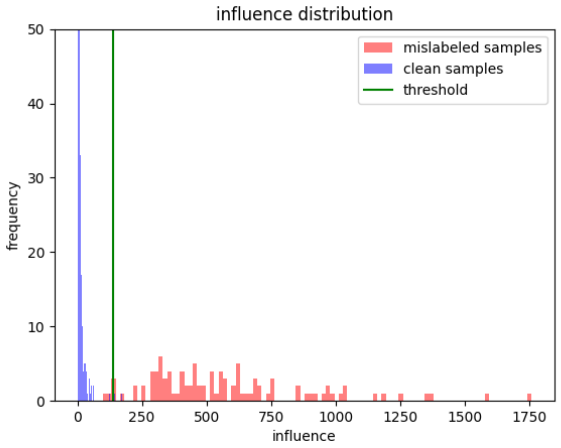

図 5.2: 混入割合 $r = 0.05$ の学習後の各サンプルに対する $I_{up,loss}(\mathcal{D}_{train} \setminus z, z)$ のヒストグラム．見やすさのため，y 軸の上限を 50 までとしている．

5.3.2 実験 2: ミスラベルデータに関連するパラメータを更新することで推論精度は向上するか

ミスラベルデータを学習済みモデルから忘却することによりモデルの推論精度が向上するか検証する．本実験ではミスラベルデータ $\mathcal{D}_{mislabeled} \subseteq \mathcal{D}$ が既知とし，$\mathcal{D}_{forget} = \mathcal{D}_{mislabeled}$ を忘却することで，推論精度が向上するか検証する．ここでは混入割合 $r = \{0.01, 0.05, 0.1, 0.2, 0.3, 0.4, 0.5\}$ とし，忘却後の性能を正解率で測ることにより，推論精度が向上するか検証する．

各忘却手法の性能を表 5.2 に示す．混入割合 r が {0.01, 0.05, 0.1} の時は，忘却前のモデル (Pre-trained) の性能がすでに高いため，忘却による性能向上はあまり見られなかった．しかし，

6：https://github.com/speed1313/mislabel-unlearning

表 5.1: **ミスラベルデータの検知性能.**

		混入割合 r						
	Method	0.01	0.05	0.1	0.2	0.3	0.4	0.5
ROC AUC (↑)	Loss	1.0	0.99	0.99	0.96	0.83	0.73	0.38
	$I_{up,loss}(z,z)$	1.0	0.99	0.99	0.96	0.83	0.72	0.36
	$I_{up,loss}(\mathcal{D}_{train},z)$	1.0	0.83	0.80	0.64	0.55	0.54	0.36
	$I_{up,loss}(\mathcal{D}_{train}\setminus z,z)$	0.95	0.71	0.75	0.60	0.54	0.53	0.37
accuracy (↑)	Loss	100	99	98	93	79	69	41
	$I_{up,loss}(z,z)$	100	99	98	92	78	67	40
	$I_{up,loss}(\mathcal{D}_{train},z)$	100	97	94	79	63	57	39
	$I_{up,loss}(\mathcal{D}_{train}\setminus z,z)$	99	97	93	77	63	55	39
precision (↑)	Loss	100	98	91	83	65	61	41
	$I_{up,loss}(z,z)$	100	96	91	81	63	59	40
	$I_{up,loss}(\mathcal{D}_{train},z)$	100	73	72	47	38	46	39
	$I_{up,loss}(\mathcal{D}_{train}\setminus z,z)$	85	66	65	43	38	44	39

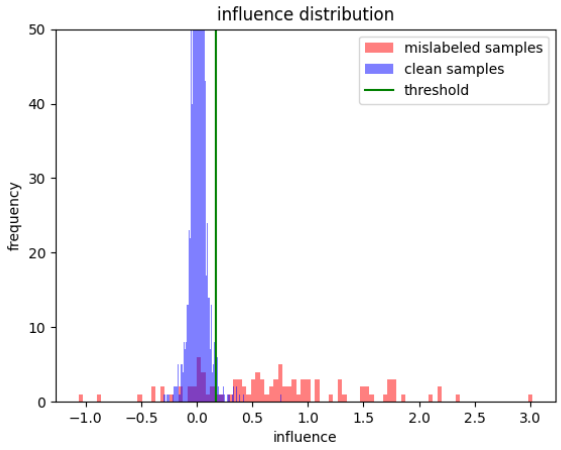

図 5.3: 混入割合 $r = 0.05$ の学習後の各サンプルに対する $I_{up,loss}(\mathcal{D}_{train},z)$ のヒストグラム. 見やすさのため, y 軸の上限を 50 までとしている.

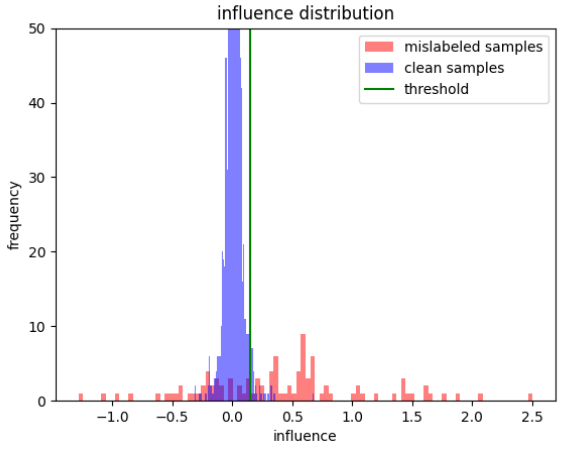

図 5.4: 混入割合 $r = 0.05$ の学習後の各サンプルに対する $I_{up,loss}(\mathcal{D}_{train}\setminus z,z)$ のヒストグラム. 見やすさのため, y 軸の上限を 50 までとしている.

混入割合 r が {0.2, 0.3, 0.4, 0.5} のとき, 忘却により推論精度が向上していることがわかる. 以上から, 混入割合 r が大きい場合, ミスラベルデータを忘却することにより, 推論精度が向上することがわかった. なお, Machine Unlearning の研究においては, 忘却対象の数が多いほど忘却後の推論精度が悪くなる傾向になることが知られている [31]. しかし, 今回は忘却対象が $\mathcal{D}_{mislabeled}$ を含んだ訓練データを用いて学習したモデルであり, 忘却前のモデルの精度が低いため, 混入割合 r が大きく, 忘却対象の数が多い場合においても忘却による精度の向上が見られたと考えられる.

5.3.3 実験 3: ミスラベルデータの検知とパラメータの更新を繋げることで推論精度は向上するか

検知したデータ $\mathcal{D}_{detected}$ を忘却の対象 $\mathcal{D}_{forget} = \mathcal{D}_{detected}$ として学習済みモデルから忘却することで, 推論精度が向上するか検証する. ここでは, ミスラベルデータの検知手法として, 実験 1 で最も良い性能を示した損失に基づく検知手法を用いることで, 忘却対象を $\mathcal{D}_{detected}$ として忘却する. 他の設定は実験 2 と同様である.

結果を表 5.3 に示す. 混入割合 r の増加に伴い, 忘却後の性能が悪化していることがわかる. また, $\mathcal{D}_{mislabeled}$ を忘却した場合に比べ, $\mathcal{D}_{detected}$ を忘却することで推論精度の向上が悪化していることがわかる. このような結果となった原因として, 混入割合 r が増加するにつれて, ミスラベルデータの検知に関する性能が低下し, ミスラベルデータとクリーンデータの入り混じったデータを忘却してしまうため, 推論精度が低下したと考えられる.

以上から, 混入割合 r が大きい場合には, ミスラベルデータの検知性能が低下するため, ミスラベルデータの検知と忘却を繋げた手法の有効性は確認できなかった. 混入割合 r が小さい場合も, 忘却前のモデルの性能が高いため, 忘却による性能向上はあまり見られなかった.

表 5.2: $\mathcal{D}_{mislabeled}$ を忘却したモデルの推論精度

		混入割合 r						
	Method	0.01	0.05	0.1	0.2	0.3	0.4	0.5
accuracy on $\mathcal{D}_{test}$ (↑)	Pre-trained	99.5	99.6	99.6	95.2	81.4	72.5	45.0
	影響関数 [18]	99.6	99.6	99.6	99.2	92.9	93.5	78.5
	SSSE [19]	97.6	99.5	97.6	98.8	95.7	93.7	93.9
	Fine-tuned	99.9	99.8	99.7	99.9	99.9	99.9	99.9
	Retrained on $\mathcal{D}_{clean}$	99.8	99.8	99.8	99.9	99.9	99.9	99.9

表 5.3: $\mathcal{D}_{detected}$ **を忘却したモデルの推論精度.** 検出手法として Loss を用いた手法を用いた. (比較対象として実験 2 の結果も載せている.)

		混入割合 r						
	Method	0.01	0.05	0.1	0.2	0.3	0.4	0.5
accuracy on $\mathcal{D}_{test}$ (↑)	Pre-trained	99.5	99.6	99.6	95.2	81.4	72.5	45.0
	影響関数 [18] on $\mathcal{D}_{mislabeled}$	99.6	99.6	99.6	99.2	92.9	93.5	78.5
	影響関数 [18] on $\mathcal{D}_{detected}$	99.6	99.6	99.6	95.7	77.0	76.2	41.2
	SSSE [19] on $\mathcal{D}_{mislabeled}$	97.6	99.5	97.6	98.8	95.7	93.7	93.9
	SSSE [19] on $\mathcal{D}_{detected}$	97.6	99.4	97.6	95.5	74.0	78.6	37.7
	Fine-tuned on $\mathcal{D}_{mislabeled}$	99.9	99.8	99.7	99.9	99.9	99.9	99.9
	Fine-tuned on $\mathcal{D}_{detected}$	99.9	99.7	99.8	97.9	85.8	76.9	40.9
	Retrained on $\mathcal{D}_{mislabeled}$	99.8	99.8	99.8	99.9	99.9	99.9	99.9
	Retrained on $\mathcal{D}_{detected}$	99.8	99.9	99.8	97.3	81.7	76.3	38.9

6 ま と め

本稿では, ミスラベルデータの忘却による学習済みモデルの汎化性能の向上手法を提案した. 提案手法は大まかには, まず学習済みモデルから計算できる異常度を通じてミスラベルデータを検知したのち, 検知したデータをモデルから忘却 (Machine Unlearning) させることで, 汎化性能を向上させる.

MNIST データセットを用いて提案手法を実験的に調査した. 詳細な分析に向けて提案手法の各段階ごとに検討したところ, まずミスラベルデータの混入割合が小さい場合に, 有効に検知できることを確認した. 次にパラメータの更新については, 混入割合によらず, 有効性が確認できた. しかし, 提案手法の構成に相当する実験として検知とパラメータの更新を繋げた場合, 忘却による機械学習モデルの性能向上が確認できなかった. これは混入割合が増加するに従い, ミスラベルデータの検知の性能が低下することに起因している. これらの機能の改善は今後の課題である.

謝　　辞

本研究の一部は JST CREST（課題番号：JPMJCR21M5）の支援を受けたものである. また, 本研究活動をご支援いただきました大阪大学・山口弘純教授に感謝いたします.

文　　献

[1] Benoît Frénay and Michel Verleysen. Classification in the presence of label noise: a survey. *IEEE transactions on neural networks and learning systems*, Vol. 25, No. 5, pp. 845–869, 2013.

[2] Gabriele Paolacci, Jesse Chandler, and Panagiotis G Ipeirotis. Running experiments on amazon mechanical turk. *Judgment and Decision making*, Vol. 5, No. 5, pp. 411–419, 2010.

[3] Viv Cothey. Web-crawling reliability. *Journal of the American Society for Information Science and Technology*, Vol. 55, No. 14, pp. 1228–1238, 2004.

[4] Octavian Suciu, Radu Marginean, Yigitcan Kaya, Hal Daume III, and Tudor Dumitras. When does machine learning FAIL? generalized transferability for evasion and poisoning attacks. In *27th USENIX Security Symposium (USENIX Security 18)*, pp. 1299–1316, Baltimore, MD, August 2018. USENIX Association.

[5] Han Xiao, Huang Xiao, and Claudia Eckert. Adversarial label flips attack on support vector machines. In *ECAI 2012*, pp. 870–875. IOS Press, 2012.

[6] Jonathan Krause, Benjamin Sapp, Andrew Howard, Howard Zhou, Alexander Toshev, Tom Duerig, James Philbin, and Li Fei-Fei. The unreasonable effectiveness of noisy data for fine-grained recognition. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part III 14*, pp. 301–320. Springer, 2016.

[7] Devansh Arpit, Stanisław Jastrzebski, Nicolas Ballas, David Krueger, Emmanuel Bengio, Maxinder S Kanwal, Tegan Maharaj, Asja Fischer, Aaron Courville, Yoshua Bengio, et al. A closer look at memorization in deep networks. In *International conference on machine learning*, pp. 233–242. PMLR, 2017.

[8] Florian Tramèr, Reza Shokri, Ayrton San Joaquin, Hoang Le, Matthew Jagielski, Sanghyun Hong, and Nicholas Car-

lini. Truth serum: Poisoning machine learning models to reveal their secrets. In *Proc. of CCS 2022*, pp. 2779–2792. ACM, 2022.

[9] Yufei Chen, Chao Shen, Yun Shen, Cong Wang, and Yang Zhang. Amplifying membership exposure via data poisoning. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, Vol. 35, pp. 29830–29844. Curran Associates, Inc., 2022.

[10] Zhibo Wang, Yuting Huang, Mengkai Song, Libing Wu, Feng Xue, and Kui Ren. Poisoning-assisted property inference attack against federated learning. *IEEE Transactions on Dependable and Secure Computing*, pp. 1–13, 2022.

[11] Giorgio Patrini, Alessandro Rozza, Aditya Krishna Menon, Richard Nock, and Lizhen Qu. Making deep neural networks robust to label noise: A loss correction approach. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1944–1952, 2017.

[12] Zhilu Zhang and Mert Sabuncu. Generalized cross entropy loss for training deep neural networks with noisy labels. *Advances in neural information processing systems*, Vol. 31, , 2018.

[13] Daiki Tanaka, Daiki Ikami, Toshihiko Yamasaki, and Kiyoharu Aizawa. Joint optimization framework for learning with noisy labels. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5552–5560, 2018.

[14] Geoff Pleiss, Tianyi Zhang, Ethan Elenberg, and Kilian Q Weinberger. Identifying mislabeled data using the area under the margin ranking. *Advances in Neural Information Processing Systems*, Vol. 33, pp. 17044–17056, 2020.

[15] Qingrui Jia, Xuhong Li, Lei Yu, Jiang Bian, Penghao Zhao, Shupeng Li, Haoyi Xiong, and Dejing Dou. Learning from training dynamics: identifying mislabeled data beyond manually designed features. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 37, pp. 8041–8049, 2023.

[16] Hugging face. `https://huggingface.co/`. Accessed: 2023-12-23.

[17] GitHub. `https.//github.com/`. Accessed: 2023-12-23.

[18] Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, Vol. 70 of *Proceedings of Machine Learning Research*, pp. 1885–1894. PMLR, 06–11 Aug 2017.

[19] Alexandra Peste, Dan Alistarh, and Christoph H Lampert. Ssse: Efficiently erasing samples from trained machine learning models. *arXiv preprint arXiv:2107.03860*, 2021.

[20] Abraham Chan, Arpan Gujarati, Karthik Pattabiraman, and Sathish Gopalakrishnan. The fault in our data stars: Studying mitigation techniques against faulty training data in machine learning applications. In *2022 52nd Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, pp. 163–171, 2022.

[21] Hwanjun Song, Minseok Kim, Dongmin Park, Yooju Shin, and Jae-Gil Lee. Learning from noisy labels with deep neural networks: A survey. *IEEE Transactions on Neural Networks and Learning Systems*, Vol. 34, No. 11, pp. 8135–8153, 2023.

[22] Yinzhi Cao and Junfeng Yang. Towards making systems forget with machine unlearning. In *2015 IEEE Symposium on Security and Privacy*, pp. 463–480, 2015.

[23] R. Shokri, M. Stronati, C. Song, and V. Shmatikov. Membership inference attacks against machine learning models. In *2017 IEEE Symposium on Security and Privacy (SP)*, pp. 3–18, Los Alamitos, CA, USA, may 2017. IEEE Computer Society.

[24] Matt Fredrikson, Somesh Jha, and Thomas Ristenpart. Model inversion attacks that exploit confidence information and basic countermeasures. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, CCS '15, p. 1322–1333, New York, NY, USA, 2015. Association for Computing Machinery.

[25] Yi Zeng, Si Chen, Won Park, Zhuoqing Mao, Ming Jin, and Ruoxi Jia. Adversarial unlearning of backdoors via implicit hypergradient. In *Proc. of ICLR 2022*. OpenReview.net, 2022.

[26] Lucas Bourtoule, Varun Chandrasekaran, Christopher A. Choquette-Choo, Hengrui Jia, Adelin Travers, Baiwu Zhang, David Lie, and Nicolas Papernot. Machine unlearning. In *2021 IEEE Symposium on Security and Privacy (SP)*, pp. 141–159, 2021.

[27] Zhiyi Tian, Lei Cui, Jie Liang, and Shui Yu. A comprehensive survey on poisoning attacks and countermeasures in machine learning. *ACM Computing Surveys*, Vol. 55, No. 8, pp. 1–35, 2022.

[28] Yinzhi Cao, Alexander Fangxiao Yu, Andrew Aday, Eric Stahl, Jon Merwine, and Junfeng Yang. Efficient repair of polluted machine learning systems via causal unlearning. In *Proc. of AsiaCCS 2018*, pp. 735–747. ACM, 2018.

[29] Junfeng Guo, Ting Wang, and Cong Liu. Poishygiene: Detecting and mitigating poisoning attacks in neural networks. *arXiv preprint arXiv:2003.11110*, 2020.

[30] Bolun Wang, Yuanshun Yao, Shawn Shan, Huiying Li, Bimal Viswanath, Haitao Zheng, and Ben Y Zhao. Neural cleanse: Identifying and mitigating backdoor attacks in neural networks. In *2019 IEEE Symposium on Security and Privacy (SP)*, pp. 707–723. IEEE, 2019.

[31] Thanh Tam Nguyen, Thanh Trung Huynh, Phi Le Nguyen, Alan Wee-Chung Liew, Hongzhi Yin, and Quoc Viet Hung Nguyen. A survey of machine unlearning. *CoRR*, Vol. abs/2209.02299, , 2022.

[32] Yann LeCun, Corinna Cortes, and CJ Burges. Mnist handwritten digit database. *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, Vol. 2, , 2010.

[33] James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018.

[34] DeepMind, Igor Babuschkin, Kate Baumli, Alison Bell, Surya Bhupatiraju, Jake Bruce, Peter Buchlovsky, David Budden, Trevor Cai, Aidan Clark, Ivo Danihelka, Antoine Dedieu, Claudio Fantacci, Jonathan Godwin, Chris Jones, Ross Hemsley, Tom Hennigan, Matteo Hessel, Shaobo Hou, Steven Kapturowski, Thomas Keck, Iurii Kemaev, Michael King, Markus Kunesch, Lena Martens, Hamza Merzic, Vladimir Mikulik, Tamara Norman, George Papamakarios, John Quan, Roman Ring, Francisco Ruiz, Alvaro Sanchez, Laurent Sartran, Rosalia Schneider, Eren Sezener, Stephen Spencer, Srivatsan Srinivasan, Miloš Stanojević, Wojciech Stokowiec, Luyu Wang, Guangyao Zhou, and Fabio Viola. The DeepMind JAX Ecosystem, 2020.

[35] Jonathan Heek, Anselm Levskaya, Avital Oliver, Marvin Ritter, Bertrand Rondepierre, Andreas Steiner, and Marc van Zee. Flax: A neural network library and ecosystem for JAX, 2023.