ⓒオペレーションズ・リサーチ

確率的自然勾配法に基づく One-Shot Neural Architecture Search

白川　真一

Neural Architecture Search (NAS) は，ニューラルネットワークの構造を自動設計する方法である．一般的な NAS では，ニューラルネットワークの学習を何度も行う必要があるため多くの計算資源を必要とする．この問題を解決するアプローチに，ニューラルネットワークの重みと構造パラメータを同時に最適化する One-Shot NAS がある．本稿では，ネットワーク構造を生成する確率分布のパラメータを確率的自然勾配法によって最適化する One-Shot NAS を紹介する．

キーワード：Neural Architecture Search，ディープニューラルネットワーク，自然勾配法

1. はじめに

ディープニューラルネットワークは画像認識や自然言語処理などで大きな成功を収めている機械学習モデルである．一方，その性能はネットワーク構造に依存する部分が大きく，これまでにさまざまな構造が開発されている．ニューラルネットワークの構造は使用者の経験や試行錯誤によって設計されるのが通常だが，その過程は使用者にとって大きな負担になると考えられる．この構造設計のプロセスを自動化しようとする方法は，Neural Architecture Search (NAS) と呼ばれ，深層学習分野において活発に研究が進められている[1]．

初期の NAS の手法は，ハイパーパラメータ最適化問題として構造最適化を定式化することがほとんどであった．つまり，構造を固定し訓練用データを使って重みを最適化した後に，検証用データに対する性能で構造の良さを評価する [1–3]．これらの手法は，高い性能を示す構造を発見できているが，数百台の GPU を用いるなど計算コストの面で課題があった．これに対して，最近の研究 [4–9] では，スーパーグラフを考え候補となる構造をそのサブグラフとして与えることで，重みと構造を 1 回の訓練中に同時に最適化する手法が提案されている．これらの手法は One-Shot NAS と呼ばれ，計算コストの大幅な削減に成功している．One-Shot NAS を実現するための有望なアプローチは，重みと構造によって定まる目的関数を，連続緩和 (Continuous Relaxation) [6, 9] もしくは確率緩和 (Stochastic Relaxation) [7, 8] によって微分可能な目的関数に変換し，勾配降下法によって最適化を行うものである．

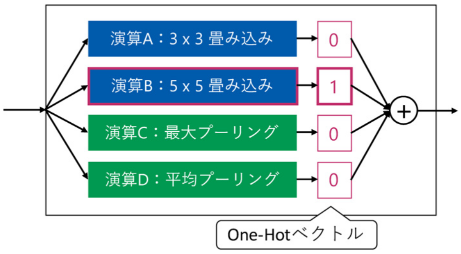

図 1　One-Hot ベクトルによる演算の選択の例．この例では，四つの候補の中から “5 × 5 畳み込み” を選択している．

ニューラルネットワークの構造最適化問題は，ネットワーク内のある層での演算の種類や層間の接続を決定する問題と考えることができる．One-Shot NAS では，ある層の演算や接続元をあらかじめ定めた候補の中から選択するものとする．この選択は One-Hot ベクトルによるカテゴリ変数などで表現することができる．図 1 は，ある層の演算を四つの候補の中から選択する例を示している．このように，構造最適化問題を冗長なネットワークの中から部分構造を選択する問題として取り扱う．このとき，各演算内に存在する学習可能なパラメータ（重みパラメータ）は従来の確率的勾配降下法によって最適化する．

連続緩和は One-Hot ベクトルをソフトマックス関数などによって連続変数に置き換える方法であり，カテゴリ変数による選択を連続変数と各演算結果の重

しらかわ　しんいち
横浜国立大学大学院環境情報研究院
〒 240–8501 神奈川県横浜市保土ヶ谷区常盤台 79–7
shirakawa-shinichi-bg@ynu.ac.jp

[1] NAS の文献リストをまとめた Web ページからもここ数年で非常に多くの論文が発表されているのが確認できる (https://www.ml4aad.org/automl/literature-on-neural-architecture-search/).

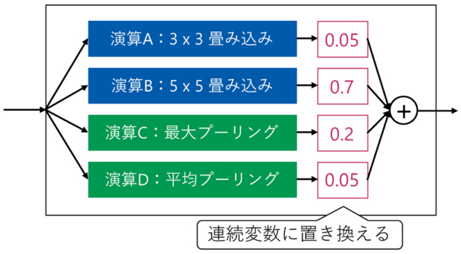

図 2　連続緩和の概念図

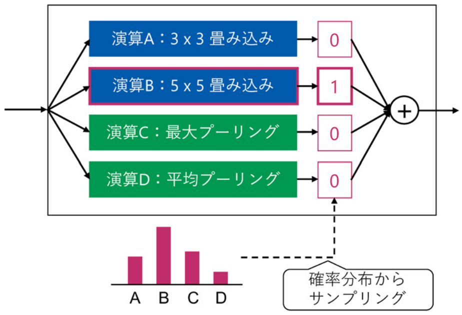

図 3　確率緩和の概念図

み付き和に置き換える．このように One-Hot ベクトルを連続変数に置き換えることで，連続変数に関する勾配計算が可能になり，これを利用して最適化を行う．学習後には，最も値が大きな変数値に対応する演算や接続が選択される．図 2 に連続緩和の概念図を示す．

一方，確率緩和ではカテゴリ変数などの構造パラメータを生成する確率分布を考え，その分布パラメータを最適化する．ここで，カテゴリカル分布を考えると，分布のパラメータは連続変数であり，勾配を計算することが可能になる．確率緩和の場合は，学習後の確率分布のパラメータを使って最も確率の高い演算や接続を選択する．図 3 に確率緩和の概念図を示す．連続緩和は勾配計算の際に候補となるすべての演算を実行する必要があるのに対して，確率緩和はサンプリングされた演算だけを実行すれば良いので，学習時の計算量とメモリ使用量の面で利点がある．

本稿では，確率緩和によって得られる微分可能な目的関数を利用し，重みと構造を勾配法によって同時に最適化する One-Shot NAS のフレームワーク [8, 10] を紹介する．この手法では，構造パラメータに対する確率分布を導入し，構造パラメータを直接最適化する代わりに確率分布のパラメータを確率的自然勾配法 [11] によって最適化する．構造パラメータの種類に応じて確率分布を選択することで，カテゴリ変数や連続変数，整数値，またはそれらの混合変数を同一のフレームワークで取り扱うことが可能になる．さらに，NAS の手法自体のチューニングの手間を削減することを目標にして開発された学習率適応アルゴリズムについても紹介する．この学習率適応機構によって学習率の設定に対して頑健な最適化が実現できる．最後に，この学習率適応機構を導入した NAS である Adaptive Stochastic Natural Gradient-Based NAS (ASNG-NAS) の有効性をいくつかの数値実験結果を通して確認する．

2.　確率的自然勾配法に基づく One-Shot NAS

2.1　基本フレームワーク

次のような最適化問題を考える．

$$\max_{\boldsymbol{x}\in\mathcal{X},\ \boldsymbol{c}\in\mathcal{C}} f(\boldsymbol{x},\boldsymbol{c}) \tag{1}$$

ここで，$f : \mathcal{X}\times\mathcal{C}\to\mathbb{R}$ は目的関数であり，$\boldsymbol{x}\in\mathcal{X}$ に関しては微分可能，$\boldsymbol{c}\in\mathcal{C}$ に関しては微分不可能であるとする．また，$\boldsymbol{x}$ の定義域 $\mathcal{X}$ は n_x 次元の実数空間 $\mathbb{R}^{n_x}$ の部分集合であるが，$\boldsymbol{c}$ の定義域 $\mathcal{C}$ はカテゴリカル，連続，もしくはそれらの直積空間であるとする．NAS の文脈では $\boldsymbol{x}\in\mathcal{X}$ がニューラルネットワークの重みパラメータ，$\boldsymbol{c}\in\mathcal{C}$ が構造パラメータに対応し，目的関数 f は負の損失関数に対応する．One-Shot NAS では，この二つの変数 $\boldsymbol{x}$ と $\boldsymbol{c}$ を，$\boldsymbol{x}$ に関する勾配 $\nabla_{\boldsymbol{x}} f$ を使いつつ，同時に最適化することを目指す．

最適化問題 (1) を確率緩和によって，微分可能な目的関数へと変換することを考える．これを実現するために，$\mathcal{C}$ 上で定義される確率分布族 $\mathcal{P}=\{P_{\boldsymbol{\theta}} : \boldsymbol{\theta}\in\Theta\subseteq\mathbb{R}^{n_\theta}\}$ を導入する．確率緩和では次のように確率分布 $P_{\boldsymbol{\theta}}$ のもとでの f の期待値を変換後の目的関数として採用する．

$$J(\boldsymbol{x},\boldsymbol{\theta}) = \mathbb{E}_{p_{\boldsymbol{\theta}}}[f(\boldsymbol{x},\boldsymbol{c})] \tag{2}$$

$$= \int_{\boldsymbol{c}\in\mathcal{C}} f(\boldsymbol{x},\boldsymbol{c})p_{\boldsymbol{\theta}}(\boldsymbol{c})\mathrm{d}\boldsymbol{c} \tag{3}$$

ここで，$p_{\boldsymbol{\theta}}$ は $P_{\boldsymbol{\theta}}$ の確率密度関数である．また，尤度関数 $\ln p_{\boldsymbol{\theta}}$ は $\boldsymbol{\theta}\in\Theta$ に関して微分可能であると仮定する．さらに，$P_{\boldsymbol{\theta}}\in\mathcal{P}$ は，任意の $\boldsymbol{c}\in\mathcal{C}$ に対して $\boldsymbol{c}$ 上の Dirac Delta 分布へと収束することができるとする．ここで，Dirac Delta 分布は，ある一点だけに確率密度をもつような分布を指す．このような設定のもと，J の最大化と元の目的関数 f の最大化は，$\sup_{\boldsymbol{\theta}\in\Theta} J(\boldsymbol{x},\boldsymbol{\theta}) = \sup_{\boldsymbol{c}\in\mathcal{C}} f(\boldsymbol{x},\boldsymbol{c}) = f(\boldsymbol{x},\boldsymbol{c}^*)$ という意味で同一になる．

確率緩和によって得られた目的関数 J は，$\boldsymbol{x}$ と $\boldsymbol{\theta}$ の両方に関して微分可能であり，$\boldsymbol{x}$ に関する勾配と $\boldsymbol{\theta}$ に関する自然勾配 [11] は次のように与えられる．

$$\nabla_{\boldsymbol{x}} J(\boldsymbol{x}, \boldsymbol{\theta}) = \mathbb{E}_{p_{\boldsymbol{\theta}}}[\nabla_{\boldsymbol{x}} f(\boldsymbol{x}, \boldsymbol{c})] \tag{4}$$
$$\tilde{\nabla}_{\boldsymbol{\theta}} J(\boldsymbol{x}, \boldsymbol{\theta}) = \mathbb{E}_{p_{\boldsymbol{\theta}}}[f(\boldsymbol{x}, \boldsymbol{c}) \tilde{\nabla}_{\boldsymbol{\theta}} \ln(p_{\boldsymbol{\theta}}(\boldsymbol{c}))] \tag{5}$$

ここで，自然勾配とはフィッシャー計量のもとでの最急方向を表し，フィッシャー情報行列の逆行列と通常の勾配の積で与えられる．

以降，確率分布族 $\mathcal{P}$ が指数型分布族である場合に限定して議論を進める．指数型分布族の確率密度関数は $h(\boldsymbol{c}) \cdot \exp(\eta(\boldsymbol{\theta})^\top T(\boldsymbol{c}) - \varphi(\boldsymbol{\theta}))$ で与えられる．ここで，$T : \mathcal{C} \to \mathbb{R}^{n_{\boldsymbol{\theta}}}$ は十分統計量，$\eta : \Theta \to \mathbb{R}^{n_{\boldsymbol{\theta}}}$ は自然パラメータ，$\varphi(\boldsymbol{\theta})$ は正規化項であり，簡単のため $h(\boldsymbol{c}) = 1$ とする．この確率分布のパラメータ $\boldsymbol{\theta}$ を $\boldsymbol{\theta} = \mathbb{E}_{p_{\boldsymbol{\theta}}}[T(\boldsymbol{c})]$ となるように選んだ場合，これを期待値パラメータと呼ぶ．この期待値パラメータのもとで，対数尤度の自然勾配は $\tilde{\nabla} \ln(p_{\boldsymbol{\theta}}(\boldsymbol{c})) = T(\boldsymbol{c}) - \boldsymbol{\theta}$ で，フィッシャー情報行列の逆行列は $\mathbf{F}^{-1}(\boldsymbol{\theta}) = \mathbb{E}[(T(\boldsymbol{c}) - \boldsymbol{\theta})(T(\boldsymbol{c}) - \boldsymbol{\theta})^\top]$ で与えられる．ベルヌーイ分布やカテゴリカル分布，ガウス分布は指数型分布族に含まれるため，ニューラルネットワークの構造パラメータの確率分布としてこれらの分布を採用することができる．

変換された目的関数 J を最大化するために，パラメータ $\boldsymbol{x}$ と $\boldsymbol{c}$ を勾配方向へと更新することを考える．勾配 (4) と (5) を解析的に得ることはできないため，独立同分布に従うサンプル $\boldsymbol{c}_i \sim P_{\boldsymbol{\theta}}$ $(i = 1, 2, \dots)$ を使ってモンテカルロ法によって推定する．変数 $\boldsymbol{x}$ に関する勾配は $\lambda_{\boldsymbol{x}}$ 個のサンプルを用いて次で推定できる．

$$G_{\boldsymbol{x}}(\boldsymbol{x}, \boldsymbol{\theta}) = \frac{1}{\lambda_{\boldsymbol{x}}} \sum_{i=1}^{\lambda_{\boldsymbol{x}}} \nabla_{\boldsymbol{x}} f(\boldsymbol{x}, \boldsymbol{c}_i) \tag{6}$$

また，$\boldsymbol{\theta}$ に関する自然勾配もサンプル $\boldsymbol{c}_i \sim P_{\boldsymbol{\theta}}$ $(i = 1, 2, \dots, \lambda_{\boldsymbol{\theta}})$ を用いて次のように推定できる．

$$G_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{\theta}) = \frac{1}{\lambda_{\boldsymbol{\theta}}} \sum_{i=1}^{\lambda_{\boldsymbol{\theta}}} f(\boldsymbol{x}, \boldsymbol{c}_i)(T(\boldsymbol{c}_i) - \boldsymbol{\theta}) \tag{7}$$

これらの勾配の推定値を用いて，交互もしくは同時に $\boldsymbol{x}$ と $\boldsymbol{\theta}$ を（自然）勾配方向に更新していくことで，目的関数 J の最適化を実現する．文献 [8] では，構造パラメータの確率分布にベルヌーイ分布を採用し，バイナリ値によって活性化関数の種類や層間の接続の有無の選択などを表現し，ニューラルネットワークの重みと構造の同時最適化ができることを示している．また，文献 [12] では，構造パラメータに関する罰則項を目的関数 f に追加することで，精度を保ちつつ構造をコンパクトにすることを可能にしている．なお，微分可能な変数 $\boldsymbol{x}$ を考えずに完全なブラックボックス関数 $f(\boldsymbol{c})$ の最適化を考えると，確率緩和を用いた自然勾配法によるブラックボックス関数の最適化は，Information Geometric Optimization (IGO) [13] として知られている．

2.2　学習率適応機構

勾配 (6) と (7) を用いた確率的自然勾配法による更新には，サンプルサイズと学習率の 2 種類のハイパーパラメータが存在する．サンプルサイズについては可能な限り大きな値をとることが望ましいが，計算コストと推定精度のトレードオフになるため利用できる計算資源に制約を受ける．一方，学習率に関しては適切な値を定める必要がある．重みパラメータの $\boldsymbol{x}$ の更新は通常のニューラルネットワークの最適化と同様であることから，Adam [14] などのアルゴリズムを利用することができる．それに対して，確率分布パラメータである $\boldsymbol{\theta}$ の更新の学習率については，ロバストな最適化を実現するためには独自の学習率適応機構が必要になる．

ここでは，推定勾配 (6) と推定自然勾配 (7) を正規化したものを用いた次のような交互の最適化を考える．

$$\boldsymbol{x}^{t+1} = \boldsymbol{x}^t + \epsilon_{\boldsymbol{x}} G_{\boldsymbol{x}}(\boldsymbol{x}^t, \boldsymbol{\theta}^t), \tag{8}$$
$$\boldsymbol{\theta}^{t+1} = \boldsymbol{\theta}^t + \epsilon_{\boldsymbol{\theta}} G_{\boldsymbol{\theta}}(\boldsymbol{x}^{t+1}, \boldsymbol{\theta}^t), \tag{9}$$
$$\epsilon_{\boldsymbol{\theta}} = \delta_{\boldsymbol{\theta}} / \|G_{\boldsymbol{\theta}}(\boldsymbol{x}^{t+1}, \boldsymbol{\theta}^t)\|_{\mathbf{F}(\boldsymbol{\theta}^t)}. \tag{10}$$

ここで，$\epsilon_{\boldsymbol{x}}$ と $\epsilon_{\boldsymbol{\theta}}$ は，それぞれ $\boldsymbol{x}$ と $\boldsymbol{\theta}$ の更新の学習率を表し，$\|G_{\boldsymbol{\theta}}\|_{\mathbf{F}(\boldsymbol{\theta})} = \sqrt{G_{\boldsymbol{\theta}}^\top \mathbf{F}(\boldsymbol{\theta}) G_{\boldsymbol{\theta}}}$ である．この $\boldsymbol{\theta}$ に関する更新は，信頼領域半径を $\delta_{\boldsymbol{\theta}}$ とした KL ダイバージェンスのもとでの信頼領域法 (Trust Region Method) と類似したものとして考えることができる．以降，この $\delta_{\boldsymbol{\theta}}$ を適応することを考えていく．

文献 [10] では，目的関数の期待値が改善されるための条件を導出し，その条件を実現するためには確率分布のパラメータの更新方向の SNR (Signal-to-Noise Ratio) が $\delta_{\boldsymbol{\theta}}$ のオーダ以上になる必要があることを示している．具体的には，式 (11) のように推定自然勾配のフィッシャー計量のもとでの SNR を一定以上に保つ必要がある[2]．

$$\frac{\|\mathbb{E}[G(\boldsymbol{\theta}^t)]\|^2_{\mathbf{F}(\boldsymbol{\theta}^t)}}{\mathrm{Tr}(\mathrm{Cov}[G(\boldsymbol{\theta}^t)]\mathbf{F}(\boldsymbol{\theta}^t))} \in \Omega(\delta_{\boldsymbol{\theta}}) \tag{11}$$

[2] 式 (11) における $\mathrm{Tr}(\cdot)$ は行列のトレース，$\mathrm{Cov}[\cdot]$ は確率変数ベクトルの分散共分散行列を表し，Ω はランダウのオーダ記法である．また，$G_{\boldsymbol{\theta}}(\boldsymbol{x}^{t+1}, \boldsymbol{\theta}^t)$ を $G(\boldsymbol{\theta}^t)$ と簡略表記している．

これを実現するために，自然勾配の累積によって推定した SNR を一定に保つように $\delta_{\boldsymbol{\theta}}$ を適応する．詳細なアルゴリズムの導出は文献 [10] を参照されたい．

2.3 学習率適応を用いた確率的自然勾配法による One-Shot NAS

2.2 節で説明した学習率適応を利用した確率的自然勾配法による One-Shot NAS を Adaptive Stochastic Natural Gradient-based NAS (ASNG-NAS) と呼んでいる．ASNG-NAS のアルゴリズムを Algorithm 1 に示す．Algorithm 1 中の 6 行目と 7 行目で学習率適応に必要な量（$\boldsymbol{s}$ と γ）を累積によって求めている．8 行目では，$\|\boldsymbol{s}\|^2/\gamma \approx \alpha$ となるように学習率の適応を行っており，これにより SNR が一定に保たれることをねらっている．

実験では, サンプルサイズを必要最小数である $\lambda_{\boldsymbol{\theta}} = 2$ として検証する．また，式 (7) 内の目的関数値から平均値を引く．これは勾配の分散低減手法としてよく知られた方法であり，勾配の期待値には影響がない．

ASNG-NAS の具体的なアルゴリズムを導出するためには, $\mathcal{C}$ 上で定義される指数型分布族を準備する必要がある．構造パラメータがカテゴリ変数の場合は，カテゴリカル分布を使用すればよく, フィッシャー情報行列およびその逆行列も解析的に与えられる．構造パラメータが連続変数や整数といった順序変数である場合は，ガウス分布 $P_{\boldsymbol{\theta}} = \mathcal{N}(\mu_1, \sigma_1^2) \times \cdots \times \mathcal{N}(\mu_{n_{\boldsymbol{c}}}, \sigma_{n_{\boldsymbol{c}}}^2)$ を用いることができる．カテゴリ変数と順序変数の直積空間を扱いたい場合は，これらの確率分布を結合した分布を利用すれば良い．

Algorithm 1 ASNG-NAS の擬似コード

Require: $\boldsymbol{x}^0$, $\boldsymbol{\theta}^0$ { 初期パラメータ }
Require: $\alpha = 1.5$, $\delta_{\boldsymbol{\theta}}^0 = 1$, $\lambda_{\boldsymbol{x}} = \lambda_{\boldsymbol{\theta}} = 2$
1: $\Delta = 1$, $\gamma = 0$, $\boldsymbol{s} = \boldsymbol{0}$, $t = 0$
2: **repeat**
3: $\delta_{\boldsymbol{\theta}} = \delta_{\boldsymbol{\theta}}^0/\Delta$, $\beta = \delta_{\boldsymbol{\theta}}/n_{\boldsymbol{\theta}}^{1/2}$
4: 式 (6) の $G_{\boldsymbol{x}}(\boldsymbol{x}^t, \boldsymbol{\theta}^t)$ を計算し，式 (8) によって $\boldsymbol{x}^{t+1}$ へ更新
5: 式 (7) の $G_{\boldsymbol{\theta}}(\boldsymbol{x}^{t+1}, \boldsymbol{\theta}^t)$ を計算し，式 (9) によって $\boldsymbol{\theta}^{t+1}$ へ更新
6: $\boldsymbol{s} \leftarrow (1-\beta)\boldsymbol{s} + \sqrt{\beta(2-\beta)} \frac{\mathbf{F}(\boldsymbol{\theta}^t)^{\frac{1}{2}} G_{\boldsymbol{\theta}}(\boldsymbol{x}^{t+1}, \boldsymbol{\theta}^t)}{\|G_{\boldsymbol{\theta}}(\boldsymbol{x}^{t+1}, \boldsymbol{\theta}^t)\|_{\mathbf{F}(\boldsymbol{\theta}^t)}}$
7: $\gamma \leftarrow (1-\beta)^2\gamma + \beta(2-\beta)$
8: $\Delta \leftarrow \min(\Delta_{\max}, \Delta \exp(\beta(\gamma - \|\boldsymbol{s}\|^2/\alpha)))$
9: **until** 終了条件を満たすまで

3. 数値実験

3.1 人工関数での実験

ASNG のロバスト性を評価するために，連続変数 $\boldsymbol{x} \in \mathbb{R}^{D \times K}$ とカテゴリ変数 $\boldsymbol{c}$ から構成される人工の目的関数を考える．ここで，カテゴリ変数 $\boldsymbol{c}$ は D 次元であり，カテゴリ数はすべての次元で K とする．各次元 i $(1 \leq i \leq D)$ に対して，$\boldsymbol{c}$ 内の i 番目のカテゴリ変数の One-Hot ベクトル表記を $h_i(\boldsymbol{c}) \in \{0,1\}^K$ と表記する．ここでは，次で定義される選択的二乗誤差関数を使用する．

$$f(\boldsymbol{x}, \boldsymbol{c}) = \mathbb{E}_{\boldsymbol{z}}\left[\sum_{i=1}^{D}\sum_{j=1}^{K} h_{ij}(\boldsymbol{c})\left((x_{ij} - z_i)^2 + \frac{j-1}{K}\right)\right]$$

ここで，$\boldsymbol{z}$ は $\mathcal{N}(0, K^{-2}I)$ に従う確率変数であるとする．この目的関数は $\boldsymbol{x}$ の有効な変数をカテゴリ変数 $\boldsymbol{c}$ によって切り替える．最適値は $h_i(\boldsymbol{c}) = [1, 0, \ldots, 0]$, $i = 1, \ldots, D$, $\boldsymbol{x}_1 = [0, \ldots, 0]$ で与えられる．最適化の際は，ニューラルネットワークの訓練を模倣して，$\boldsymbol{z}$ をデータサンプルと見立て，各パラメータ更新で $\mathcal{N}(0, K^{-2}I)$ からサンプルされる $\boldsymbol{z}$ で期待値を近似する．

連続変数 $\boldsymbol{x}$ を最適化するために慣性項付きの確率的勾配降下法を利用し，慣性項の係数を 0.9 とし，学習率 $\epsilon_{\boldsymbol{x}}$ をコサインスケジューリング [15] で減衰させる．連続変数を $\boldsymbol{x} \sim \mathcal{N}(0, I)$，確率分布パラメータを $\boldsymbol{\theta} = (1/K)\mathbf{1}$ で初期化する．更新回数 10^5 以内に目的関数値が $K^{-1} + DK^{-2}$ 未満となる解が得られた場合に最適化成功とみなす．100 回の試行に対して最適化成功時の更新回数の中央値を成功率で割ったものを性能指標とし，ここでは典型的な結果として $D = 30$, $K = 5$ の場合の結果を示す．

図 4 は，初期学習率 $\delta_{\boldsymbol{\theta}}^0$ を変化させたときの ASNG, SNG（一定の学習率を用いた確率的自然勾配法）, Adam [14] を比較した結果である．SNG と Adam では高い性能を得るためには学習率の調整が必要であり，設定が適切でないと最適化に失敗していることがわかる．それに対して，ASNG では初期学習率 $\delta_{\boldsymbol{\theta}}^0$ の影響を緩和することに成功している．

3.2 画像分類のための CNN の構造探索

一般画像分類タスクは多くの NAS の評価指標として利用されるベンチマークである．ここでは，CIFAR-10 データセット [16] を利用して既存の NAS の手法と性能比較を行った結果を紹介する．実験設定は，文献

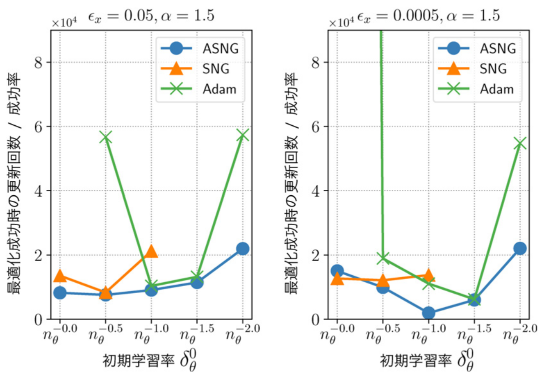

図 4 人工関数における結果の比較（$\epsilon_{\boldsymbol{x}} = 0.05$（左）と 0.0005（右）の場合）．プロットがない設定は 1 度も成功しなかったことを表している．

[6, 7] に従うものとしている．構造探索時には，訓練データを二つに分割 ($\mathcal{D} = \{\mathcal{D}_{\boldsymbol{x}}, \mathcal{D}_{\boldsymbol{\theta}}\}$) し，勾配 (6) と (7) の計算に，$\mathcal{D}_{\boldsymbol{x}}$ と $\mathcal{D}_{\boldsymbol{\theta}}$ からのミニバッチをそれぞれ使用する．このように重みと構造パラメータの最適化に異なる訓練データを用いるのは，構造最適化による過学習を防ぐためである．

ネットワーク構造の探索空間は文献 [7] で利用されているものを用いる．この探索空間では，セルと呼ばれる構造が最適化対象であり，セルの繰り返しにより CNN 全体の構造を表現する．セルは標準セルと縮小セルの 2 種類を用意し，複数の標準セルの後に縮小セルをおく構造を繰り返すことで CNN 全体を表現する．CNN の各セルは入力ノード二つ，中間ノード五つ，出力ノード一つで構成し，中間ノードの入力と演算を最適化の対象とする．各入力ノードはそれぞれ直前のセルの出力，二つ前のセルの出力を受け付ける．各中間ノードは入力を二つ受け取り，選択された演算を各入力にかけた後，演算結果を加算して出力する．ノードの入力の選択候補は入力ノードおよび既に入力が決定されたノードであり，演算は，1) 処理なし，2) 3×3 separable convolution, 3) 5×5 separable convolution, 4) 平均プーリング，5) 最大プーリングの五つから選択される．この設定では構造の組み合わせは約 2.56×10^{25} となる．また ASNG-NAS が利用する確率分布として，カテゴリカル分布を採用する．構造探索フェーズでは，$\boldsymbol{x}$ と $\boldsymbol{\theta}$ を 100 エポック分更新する．構造探索後に，最も確率の高い構造 $\hat{\boldsymbol{c}} = \mathrm{argmax}_{\boldsymbol{c}}\, p_{\boldsymbol{\theta}}(\boldsymbol{c})$ を選び，最初から重みパラメータのみを訓練し直す．その他の詳細な実験設定は文献 [10] を参照されたい．

ASNG-NAS と既存手法の探索時間とテストエラーを表 1 に示す．探索時間は使用した GPU 台数と計算日数の積である GPU days で表記されており，ASNG-NAS は 0.11GPU days (2.6 GPU hours) で構造探索を完了している．表中の下の五つの手法は探索空間が類似している手法であり，性能差は最適化アルゴリズムに起因するものと考えられる．表 1 の結果から，探索時間と最終性能にはトレードオフがあることが読み取れる．NASNet-A や NAONet は ASNG-NAS に比べると高い性能を達成しているが，数千倍以上の時間を要している．ASNG-NAS とほかの One-Shot NAS (ENAS, DARTS, SNAS) を比較すると，ASNG-NAS は最も高速に同程度の性能を示す構造の探索に成功している．

表 1 一般画像認識タスクでの比較 (CIFAR-10) [10]

手法名	探索時間 (GPU days)	テスト誤差 (%)
NASNet-A [17]	1,800	2.65
NAONet [18]	200	2.11
ProxylessNAS-G [5]	4	2.08
SMASHv2 [4]	1.5	4.03
DARTS [6] (second order)	4	2.76
DARTS [6] (first order)	1.5	3.00
SNAS [9]	1.5	2.85
ENAS [7]	0.45	2.89
ASNG-NAS	0.11	2.83

4. おわりに

本稿では，確率緩和に基づく One-Shot NAS の手法を紹介した．特に，構造の確率分布パラメータの最適化に用いる確率的自然勾配法の学習率適応機構を導入した ASNG-NAS について説明を行った．数値実験によって ASNG-NAS の探索の速さとロバスト性，性能の良さを確認した．今回の実験では，サンプルサイズ（$\lambda_{\boldsymbol{x}}$ および $\lambda_{\boldsymbol{\theta}}$）を 2 に固定したが，並列計算機が使用できる環境においては，サンプルサイズを増やすことによって，大きな学習率を取ることができるようになり，さらなる効率化が見込める．ASNG-NAS をはじめとした One-Shot NAS では，重みパラメータの学習率や正則化係数などの学習プロセスに関わるハイパーパラメータの最適化は行えない．しかし，構造パラメータの最適化が高速にできることから，ベイズ最適化などのハイパーパラメータ最適化法と One-Shot NAS を組み合わせてニューラルネットワークの構造と学習に関わるハイパーパラメータを最適化することも可能である．

謝辞　RAMP シンポジウムでの講演の機会をくださった前原貴憲氏，本稿執筆のお誘いをくださり原稿にコメントをくださった高野祐一氏に感謝いたします．また，本稿で紹介した研究成果の共著者である秋本洋平氏をはじめとした共同研究者の皆様に感謝いたします．

参考文献

[1] E. Real, S. Moore, A. Selle, S. Saxena, Y. L. Suematsu, J. Tan, Q. V. Le and A. Kurakin, "Large-scale evolution of image classifiers," In *International Conference on Machine Learning (ICML)*, pp. 2902–2911, 2017.

[2] M. Suganuma, S. Shirakawa and T. Nagao, "A genetic programming approach to designing convolutional neural network architectures," In *Genetic and Evolutionary Computation Conference (GECCO)*, pp. 497–504, 2017.

[3] B. Zoph and Q. V. Le, "Neural architecture search with reinforcement learning," In *International Conference on Learning Representations (ICLR)*, 2017.

[4] A. Brock, T. Lim, J. M. Ritchie and N. Weston, "SMASH: One-shot model architecture search through hypernetworks," In *International Conference on Learning Representations (ICLR)*, 2018.

[5] H. Cai, L. Zhu and S. Han, "ProxylessNAS: Direct neural architecture search on target task and hardware," In *International Conference on Learning Representations (ICLR)*, 2019.

[6] H. Liu, K. Simonyan and Y. Yang, "DARTS: Differentiable architecture search," In *International Conference on Learning Representations (ICLR)*, 2019.

[7] H. Pham, M. Y. Guan, B. Zoph, Q. V. Le and J. Dean, "Efficient neural architecture search via parameter sharing," In *International Conference on Machine Learning (ICML)*, pp. 4095–4104, 2018.

[8] S. Shirakawa, Y. Iwata and Y. Akimoto, "Dynamic optimization of neural network structures using probabilistic modeling," In *AAAI Conference on Artificial Intelligence (AAAI)*, pp. 4074–4082, 2018.

[9] S. Xie, H. Zheng, C. Liu and L. Lin, "SNAS: Stochastic neural architecture search," In *International Conference on Learning Representations (ICLR)*, 2019.

[10] Y. Akimoto, S. Shirakawa, N. Yoshinari, K. Uchida, S. Saito and K. Nishida, "Adaptive stochastic natural gradient method for one-shot neural architecture search," In *International Conference on Machine Learning (ICML)*, pp. 171–180, 2019.

[11] S. Amari, "Natural gradient works efficiently in learning," *Neural Computation*, **10**, pp. 251–276, 1998.

[12] S. Saito and S. Shirakawa, "Controlling model complexity in probabilistic model-based dynamic optimization of neural network structures," In *International Conference on Artificial Neural Networks (ICANN)*, pp. 393–405, 2019.

[13] Y. Ollivier, L. Arnold, A. Auger and N. Hansen, "Information-geometric optimization algorithms: A unifying picture via invariance principles," *Journal of Machine Learning Research*, **18**, pp. 564–628, 2017.

[14] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," In *International Conference on Learning Representations (ICLR)*, 2015.

[15] I. Loshchilov and F. Hutter, "SGDR: Stochastic gradient descent with warm restarts," In *International Conference on Learning Representations (ICLR)*, 2017.

[16] A. Krizhevsky, "Learning multiple layers of features from tiny images," Technical Report, 2009.

[17] B. Zoph, V. Vasudevan, J. Shlens and Q. V. Le, "Learning transferable architectures for scalable image recognition," In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8697–8710, 2018.

[18] R. Luo, F. Tian, T. Qin, E. Chen and T. Liu. "Neural architecture optimization," In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 7827–7838, 2018.

