修士論文

Softmax関数を用いたDeep Metric Learningにおける動的スケーリングパラメタに関する研究

令和3年1月28日 提出

指導教員　工藤　知宏 教授

東京大学大学院
工学系研究科　電気系工学専攻
37-186463　　佐藤　啓樹

要旨

Softmax 関数を用いる Metric Learning の学習は、特徴量空間中の距離指標として用いられる cosine 類似度に乗じるスケーリングパラメタによって大きく左右される。本稿では、学習の進展に伴う Recall@1 スコアの変化をスケーリングパラメタを変えながら測定し、最適な値が理論的に予測される値から大きくずれることを示した。また、スケーリングパラメタが cosine 類似度の差を強調する性質を持つことに着目し、学習の過程でスケーリングパラメタの値を変更することで Recall@1 スコアが 1〜2 ポイント改善することを示した。さらに広範な実験から、スケーリングパラメタが学習に与える影響について考察を行った。

Abstract

Metric learning using the Softmax function is greatly influenced by the scaling parameter that multiplies the cosine similarity used as a distance measure in the feature space. In this paper, we measure the change of Recall@1 score with the progress of learning while changing the scaling parameter, and show that the optimal value deviates significantly from the theoretically predicted value. We also note that the scaling parameter has the property of emphasizing the difference between cosine similarities, and show that changing the value of the scaling parameter during the learning process improved the Recall@1 score by 1 to 2 points. From a wide range of experiments, we also discussed the effect of scaling parameters on metric learning using Softmax function.

目 次

第 1 章　序論 **1**

1.1　研究背景 1

1.2　研究目的 3

1.3　本研究の貢献 3

1.4　本論文の構成 4

第 2 章　関連研究 **5**

2.1　Deep Learning による画像認識モデル 5

2.1.1　Convolutional Neural Network 5

2.1.2　発展的な CNN 6

2.2　Metric Learning 7

2.3　Deep Metric Learning 9

2.3.1　画像認識における Deep Metric Learning 9

2.3.2　損失関数 10

2.3.3　発展的な Deep Metric Learning 11

第 3 章　分類型ネットワークで行う Metric Learning とスケーリングパラメタ **14**

3.1　代表点とクラス分類で行う Metric Learning 14

3.1.1　proxy: 各クラスの代表点 14

3.1.2　Cross Entropy Loss を応用した Metric Learning 15

3.2　スケーリングパラメタ 16

3.2.1　cosine 類似度でクラス分類を行う問題点 16

3.2.2　新しいパラメタの導入 17

3.2.3　AdaCos 19

3.2.4　提案手法 20

第 4 章　減少するスケーリングパラメタを導入した追加学習　22
4.1　実験条件 22
4.1.1　データセット 22
4.1.2　評価指標 22
4.1.3　損失関数 23
4.1.4　その他実験条件 25
4.2　実験 26
4.2.1　実験 1: 最適な定数スケーリングパラメタ 26
4.2.2　実験 2: 線形減少するスケーリングパラメタを用いた追加学習 . . . 29
4.2.3　実験 3: 既存手法との比較 29

第 5 章　スケーリングパラメタが学習に与える影響　33
5.1　学習中におけるクラス代表点間の角度の変化 33
5.2　通常学習時にスケーリングパラメタを線形に減少させる手法 35
5.3　2 次関数的に減少するスケーリングパラメタ 37

第 6 章　結論　39
6.1　結論 39
6.2　今後の展望 40

謝辞　41

発表文献　42

付 録 A 付録　49

図 目 次

1.1 データの類似度に基づく空間や変換の概念図。 2
1.2 角度を距離指標としてクラス同士が離れて分布する特徴量空間。 3

2.1 CNN アーキテクチャの例 [1]。 6
2.2 GoogleNet を構成する Inception モジュール。 7
2.3 ResNet で用いられる Residual Block。 7
2.4 ミンコフスキ距離のノルム p によって定められる単位円 [2]。 8
2.5 顔認識を例とする Deep Metric Learning[3]。 10
2.6 Triplet Loss。 11
2.7 N-pair Loss。 12
2.8 特徴量空間で学習に有効な位置を計算し、データ点として生成する手法 [4]。 13

3.1 Proxy-NCA。 15
3.2 温度パラメタによって変化する 2 クラス間の決定境界 [5]。 17
3.3 クラスベクトルとデータベクトルがなす角度および両者が同一クラスである確率について、スケーリングパラメタの値を変化させた時の閾値の変化 [6]。 18
3.4 スケーリングパラメタとマージンによる確率の閾値の変化 [3]。 18
3.5 同一のベクトルに異なるスケーリングパラメタ s を乗じて Softmax 関数に入力した時の出力の変化。(a)$s = 1$、(b)$s = 2$、(c)$s = 5$、(d)$s = 10$ 21

4.1 CUB-200-2011 データセット [7]。 23
4.2 Cars-196 データセット [8]。 23
4.3 同一クラスで見た目が異なるデータの例 [9]。 24
4.4 SoftTriple Loss の全結合層部 [9]。 24
4.5 定数スケーリングパラメタで学習した時の Recall@1 スコアの変化。(a)CUB-200-2011、(b)Cars-196 27

4.6 スケーリングパラメタを線形に減少させながら追加学習を行った実験における損失関数の値、分類精度、Recall@1 スコアの変化。(a)CUB-200-2011、(b)Cars-196 30
4.7 異なる手法でスケーリングパラメタを減少させた実験における損失関数の値、分類精度、Recall@1 スコアの変化。(a)CUB-200-2011、(b)Cars-196 . 31

5.1 クラスの代表点ベクトルを初期化した際の cosine 類似度の分布。 34
5.2 CUB-200-2011 データセットの学習用クラスについて、スケーリングパラメタ s と特徴量次元数 d を変えて学習した時の代表点同士の cosine 類似度の分布変化。(a)1epoch、(b)20epoch、(c)40epoch、(d)50epoch 34
5.3 通常学習と追加学習それぞれでスケーリングパラメタを減少させた実験における損失関数の値、分類精度、Recall@1 スコアの変化。(a)CUB-200-2011、(b)Cars-196 36
5.4 下に凸な二次関数に従って減少するスケーリングパラメタを用いた学習結果。(a) 損失関数、(b)accuracy、(c)Recall@1 38

A.1 GoogleNet アーキテクチャの全体図。 49
A.2 VGG アーキテクチャと比較した ResNet アーキテクチャの全体図。 50
A.3 線形減少するスケーリングパラメタで追加学習を行った際の、特徴量次元数 64(上段) と 512(下段) それぞれにおける cosine 類似度の分布変化。(a)75epoch、(b)100epoch 51

表 目 次

3.1 損失関数に対する計算量のオーダ比較。 15

4.1 実験 1 における学習終了時の Recall@1 スコア。 27
4.2 Cars-196 データセットを用いて Proxy Anchor Loss[10] で学習を行った時の、スケーリングパラメタとマージンによる評価スコアの違い。 28
4.3 実験 2 における通常学習と追加学習終了時の分類精度と Recall@1 スコア。 29
4.4 実験 3 における通常学習終了時と追加学習終了時の分類精度と Recall@1 スコア (CUB-200-2011)。 32
4.5 実験 3 における通常学習終了時と追加学習終了時の分類精度と Recall@1 スコア (Cars-196)。 32

5.1 CUB-200-2011 データセットの学習用クラスについて、スケーリングパラメタ s と特徴量次元数 d を変えて学習した時の代表点同士の cosine 類似度の分散の変化。 35
5.2 追加実験 2 における通常学習と追加学習終了時の分類精度と Recall@1 スコア。 37

第1章　序論

1.1　研究背景

近年、Deep Learning を用いた画像認識に関する研究が盛んに行われている [11, 12, 10]。画像認識には、データに含まれるクラスが学習時と推論時で異なるタスクがあり、顔認証や人物同定などがその代表例である。これらのタスクに対してモデルを学習用データの分類器として学習を行うと、推論環境で未知のクラスが出てきた時に認識することができない。例えば顔認証モデルを学習させる際、学習用データを分類できるようになるだけでは推論環境でも既知の顔しか判別できないという課題がある。

Metric Learning は、学習用データで学習したモデルが一般的なデータの分布をよく捉えているという仮定のもと、学習データには無いクラスについても推論時に適切な距離指標を与えることが大きな特徴の一つである手法である。2 次元平面での Metric Learning の模式図を図 1.1 に示す。

近年 Deep Learning の代表的なモデルである DNN(Deep Neural Network) を用いて画像などの高次元データからより良い特徴量を抽出することが可能となり、これを用いた Deep Metric Learning に関する研究が数多く提案されている [13, 11, 9, 14]。特徴量の抽出に DNN モデルを用いる際には、タスクやデータに応じた損失関数を設計して逆誤差伝播法で学習を行う。データの特徴量を高次元空間内の点と捉える Metric Learning の考え方に基づいて、ユークリッド距離などの距離指標で算出したデータ間類似度を用いる損失関数が数多く提案されている [15, 16]。代表的な損失関数として、画像のペアをネットワークに入力し、得られた 2 つの特徴量点が同じクラスなら近く (2 点間の距離が小さく)、異なるクラスなら遠く (距離が大きく) なるように特徴抽出モデルを学習する方法が挙げられる [17]。つまりラベルを元にデータや特徴量同士の距離を学習し、異なるクラスが遠くにマッピングされるような空間や変換を獲得する。推論時は学習時と同様にデータを特徴量空間に埋め込み、その空間に適切な閾値を設けることでクラス分類や異常検知などが可能となる。この学習法の発展形として、一度に 3 つのデータを用いて損失関数を計算する研究が提案されている [16]。あるデータ anchor に対し、anchor と同じクラスのデータ positive と異な

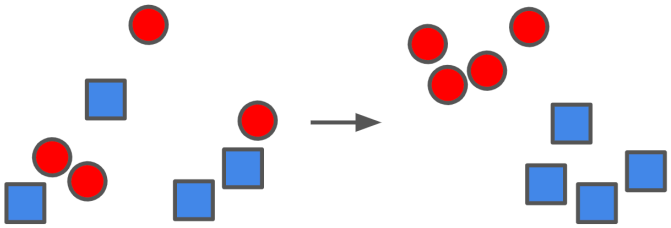


図 1.1　データの類似度に基づく空間や変換の概念図。

るクラスのデータ negative を用意し、anchor と positive のデータ同士はより近く、anchor と negative のデータ同士はより離れて位置するように特徴抽出モデルの学習を行う。

これらの研究が抱える大きな課題の一つとして、データ数 N に対してデータの組み合わせ方が N^2 や N^3 と膨大になり学習が進みにくいことが挙げられる。これに対し、学習に効果的なデータの組み合わせをミニバッチ内から探し出す hard sample mining[18, 19] や、データセットの各クラスについて中心点や代表点 (proxy) のみに損失関数を適用して学習する手法 [20] が提案されている。hard sample とは学習途中のモデルで識別が困難な、クラスの決定境界付近に位置するデータを意味する。逆に容易に識別できて損失関数が小さくなるデータは easy sample と呼ばれる。また別のアプローチとして、最終層に学習データのクラス数に応じた全結合層を接続して Softmax 関数と Cross Entropy Loss のような損失関数を用いるクラス分類型のアーキテクチャで特徴量抽出モデルを獲得する手法も多数提案されている [11, 21, 9, 20]。

また、データから得る特徴量ベクトルやクラス分類型アーキテクチャで使用する全結合層の重みを L2 正規化することで、特徴量空間を球面上に制限する手法が有効であることが知られている [22, 23]。さらにこの応用として、角度を類似度の指標とみなす手法が有効であることが知られている [11, 24, 25, 26]。図 1.2 に 2 次元球面での Metric Learning の模式図を示す。重み W はクラスの代表点であり、その近傍に属するデータが集まるような変換として Deep Neural Network を学習する。クラス間のマージン m は、クラス同士の決定境界をより厳密にしクラス内分散を小さくするために設けられる。

しかしこの cosine 類似度は値が-1 から 1 の範囲に限られており、これをそのまま Softmax 関数に入力すると正解クラスとの類似度が高く判定できても他クラスの類似度との差がほとんど無くなってしまう。そのため、最終の全結合層には cosine 類似度を定数倍してから入力する。この定数は、Neural Network モデルの知識蒸留 [27] においては温度 (Temperature) パラメタ、Deep Metric Learning においてはスケーリングパラメタと呼ばれる。Metric Learning の評価スコアである Recall@K は、このスケーリングパラメタの取り方に大きく

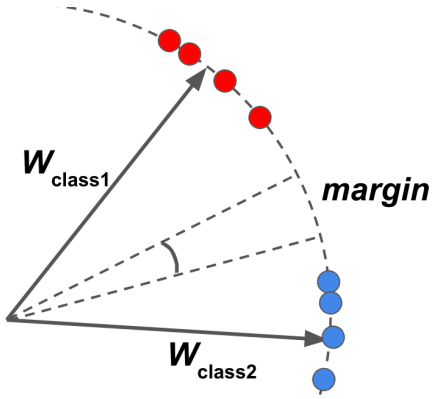


図 1.2　角度を距離指標としてクラス同士が離れて分布する特徴量空間。

依存するが、既存の研究ではこれを経験的に決めている。顔認証の分野などでは、データセットのクラス数や学習の進み具合に応じて最適なスケーリングパラメタを導出する試みがなされているが [6, 28]、論文中ではクラス数が数千オーダー以上の顔認証データセットでのみ有効性が確認されており、クラス数が数百オーダーのケースに適用すると、経験値とはかけ離れた値が算出される。

1.2　研究目的

本論文ではこのスケーリングパラメタの適切な取り方について、学習中に損失関数、分類精度、評価スコア Recall@1 を計測しながら考察する。特に、通常の学習後にスケーリングパラメタを減少させながら追加で学習する手法について提案する。

1.3　本研究の貢献

学習の進展に伴う Recall@1 スコアの変化をスケーリングパラメタを変えながら測定し、クラス数を用いて最適な値を算出する既存手法の値から大きくずれることを示した。また、スケーリングパラメタが cosine 類似度の差を強調する性質を持つことに着目し、学習の過程でスケーリングパラメタの値を変更することで Recall@1 スコアが 1〜2 ポイント改善することを示した。さらに広範な実験から、スケーリングパラメタが Softmax 関数を用いた学習に与える影響について考察を行った。

1.4　本論文の構成

2 章では本論文の背景となる関連研究について述べる。3 章では本論文の対象である分類型ネットワークアーキテクチャで Metric Learning を行う手法と、提案手法である動的なスケーリングパラメタについて説明する。4 章で実験内容と諸条件について説明し、結果と考察を述べ提案手法の有効性を確認する。5 章ではさらに実験を行い、スケーリングパラメタが学習に与える影響について考察する。最後に 6 章で結論と今後の展望について述べる。

第2章　関連研究

2.1　Deep Learningによる画像認識モデル

2.1.1　Convolutional Neural Network

CNN(Convolutional Neural Network) は、近年の画像認識分野において多くのタスクでベンチマークスコアを更新している手法である。図 2.1 に、CNN モデルである LeNet[1] のアーキテクチャを示す。

CNN は畳み込みとプーリングと呼ばれる処理を繰り返して、高次元データである画像から特徴量ベクトルを獲得する多層構造のニューラルネットワークである。畳み込み (convolution) 層では、フィルタやカーネルと呼ばれる 1×1 や 3×3 の行列を入力に対して畳み込み演算することで特徴量マップを得る。畳み込み演算とは多くの場合、入力行列と重み行列の要素積をとってから足し合わせる操作を表す。プーリング (pooling) 層では前の層で得た特徴量マップに対してフィルタリングを行い、サイズを圧縮する。フィルタリングには 2×2 や 3×3 の要素から最大値や平均値を求める Max Pooling や Average Pooling が用いられることが多い。

畳み込みとプーリングを繰り返して得たベクトルを特徴量ベクトルと呼ぶ。この次元数は数十から数百次元にすることが多く、これらは入力した画像に含まれる特徴量を表している。図 2.1 にもある通り、この後段には全結合 (fully connection) 層と呼ばれる層が接続されている。例として教師あり学習でクラス分類を行う際には最終出力の次元数がデータセットのクラス数と等しくすることで、入力画像をクラス数の次元に変換することができる。これらの出力は最後に Softmax 関数に入力され、0 から 1 の値に圧縮される。Softmax 関数は以下の式で表される。

$$f_{softmax}(x_i) = \frac{\exp x_i}{\sum_i \exp x_i}$$

この式は入力されたベクトルについて、要素全体の和が 1 になるように変換するため、画像分類においては各次元が「そのクラスに属する確率」を表していると考えることができる。

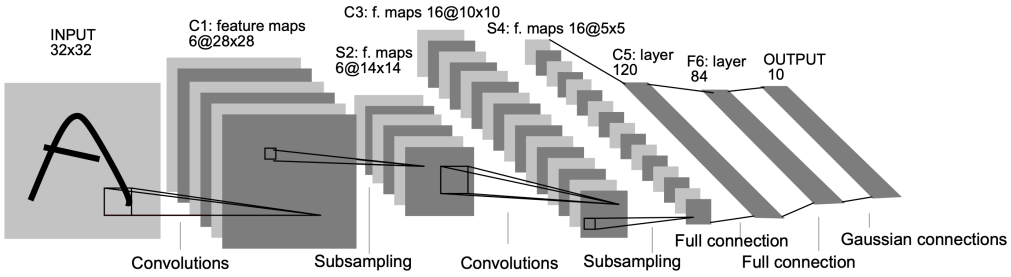


図 2.1　CNN アーキテクチャの例 [1]。

ネットワークの学習に用いる損失関数には下式で表される Cross Entropy Loss などがある。

$$l_{CE} = \sum_i t_i \log y_i$$

Softmax 関数によって Neural Network の出力は各クラスと予想される確率のベクトルになっており、この式で損失を計算する。つまり、正解ラベルに対応する出力値が大きな値であれば損失関数の値は小さく、逆に間違ったクラスに対応する出力値が大きいと損失関数の値は大きくなる。この損失関数の値をネットワークの持つ重みで微分し逆誤差伝播による確率的勾配降下法 [29] などによって CNN 全体の重みを更新することで学習を進める。

2.1.2　発展的な CNN

CNN は、畳み込み層とプーリング層をより多く重ね合わせることでより有効な特徴量を獲得できることが知られている [30]。GoogleNet[31] は、畳み込み層やプーリング層を組み合わせた Inception モジュールとよばれる小さなモジュールを多段に重ね合わせた構成を持つ CNN である。本研究における実験や比較対象とする既存手法 [6, 28] でも、このモジュールに Batch Normalization[32] を加えたアーキテクチャを使用している。図 2.2 に Inception モジュールを示す。(a) が畳み込みのみ、(b) は次元削減を同時に行う Inception モジュールである。また、付録の図 A.1 に全体のアーキテクチャ構成を示す。最終段の全結合層を、次元数が各学習用データセットのクラス数に対応させて学習を行う。

しかし、損失関数を計算し逆誤差伝播法によりネットワークの学習を行う手法を用いる際には、入力に近い層ほど勾配が大きくなりすぎたり (勾配爆発)、逆に小さくなりすぎる (勾配消失) といった課題がある。この課題を解決する手法として Residual Connection や Skip Connection と呼ばれる構造を取り入れた ResNet[12] が提案されており、本研究で扱う Deep Metric Learning でも広く用いられている。図 2.3 に、Residual Connection を含

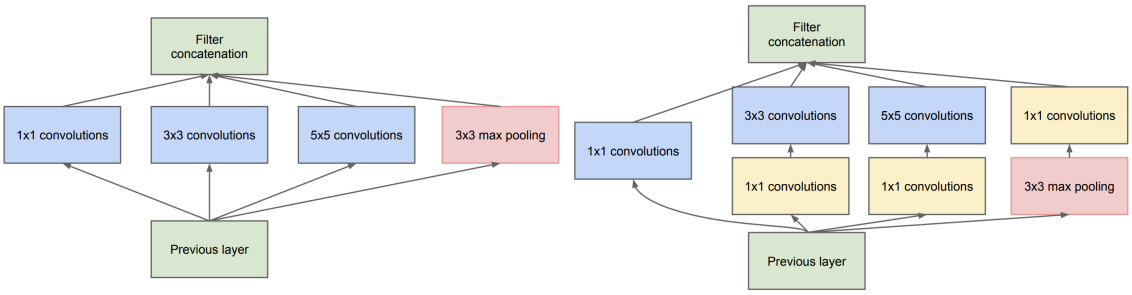


図 2.2　GoogleNet を構成する Inception モジュール。

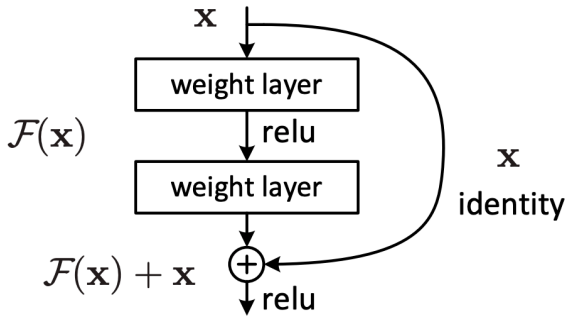


図 2.3　ResNet で用いられる Residual Block。

むブロックの構成を示す。また、付録の A.2 に全体のアーキテクチャ構成を示す。このように通常の接続に加えていくつかの層をスキップした接続を取り入れることで、forward 時には局所的な特徴量と大域的な特徴量の両方を保持して予測を行うことができると同時に、逆誤差伝播時には Residual Connection を通して入力に近い層にも出力層に近い勾配を渡すことができる。この Residual Connection がネットワーク全体の学習を容易にしている。

2.2　Metric Learning

Metric Learning は、データ間の計量距離や類似度などを学習する手法である [2, 13]。直感的には、データの次元空間あるいは特徴量空間において意味的に近いデータ同士が近く、意味の遠いデータ同士が遠くなるような変換や空間を獲得する。データ間の意味的な距離をもとに特徴量学習を行うことができ、教師ラベルの付け方で特徴量空間、つまり考慮したいデータの意味をある程度コントールできる利点がある。情報検索、ランキング、異常検知、クラス分類など多くの応用が提案されており、未知クラスのデータに対してもある程度頑健に対応できる点から本研究のような画像分類などの応用が研究されている。

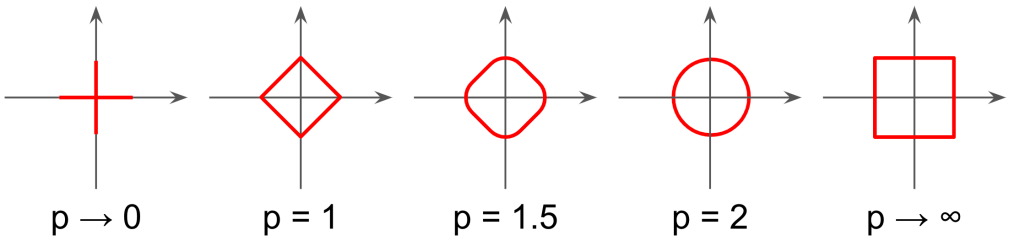


図 2.4　ミンコフスキ距離のノルム p によって定められる単位円 [2]。

距離の指標や空間には様々なものが用いられる。ミンコフスキ距離は以下の式で表され、$L_p(p \geq 1)$ ノルムとも表される。

$$\begin{aligned} d_{Minkowski}(x, y) &= ||x - y||_p \\ &= (\sum_{i=1}^{d} |x_i - y_i|^p)^{\frac{1}{p}} \end{aligned}$$

p の値によって変化する単位円の様子を図 2.4 に示す。このうち $p = 1, 2, \infty$ の時にはそれぞれマンハッタン距離、ユークリッド距離、チェビシフ距離と呼ばれ、距離指標を表す際に広く用いられている。

また、マハラノビス距離 [33, 34] も広く用いられる距離指標である。マハラノビス距離の学習では、同じ次元を持つベクトル x と y の間の距離指標行列を獲得するために、以下の式で表される共分散行列 M を学習する。

$$d_M(x, y) = \sqrt{(x - y)^T M (x - y)}$$

マハラノビス距離はデータの各次元で分散を考慮した一般的な距離を表していることになり、M の要素がすべて 1 であればユークリッド距離となる。M が半正定値行列であればさらに変形可能である。

$$\begin{aligned} d_M(x, y) &= \sqrt{(x - y)^T M (x - y)} \\ &= \sqrt{(x - y)^T L^T L (x - y)} \\ &= ||Lx - Ly||_2 \end{aligned}$$

つまり、データ間のユークリッド距離が適切になるような変換 L や、特徴量 Lx、Ly の学習を意味する。これは計量を特徴量 (特徴量空間) や特徴量抽出のための関数を学習していると考えることもできる。M や L の決め方としては、以下の式で表される最適化問題を解

けばよい。

$$\begin{aligned} \underset{M}{\text{maximize}} \quad & \sum_{(x_i,x_j)\in D} (d_M(x_i,x_j)) \\ \text{subject to} \quad & \sum_{(x_i,x_j)\in S} (d_M^2(x_i,x_j)) \leq 1, \\ & M \geq 0 \end{aligned}$$

なおここでは、類似データの組 S と非類似データの組 D をラベルつきデータセットとして用意しておく必要がある。同じクラスのデータを類似、異なるクラスのデータを非類似として扱う条件設定などが適用される。このように似ているデータが近く、似ていないデータは遠くなるような変換を学習させるのが Metric Learning の特徴である。この距離指標という特徴を利用し、上述の検索や異常検知といった特定のタスクだけでなく、特徴量空間において Adversarial Example を正解クラスに近づける学習を行うことでモデルを頑健にする手法 [35] や、データ生成器と識別器を同時に学習させる GAN に応用することで学習用データに無いクラスのデータを生成する手法 [36] など近年の深層学習が抱える課題に対しても様々な応用が研究・提案されている。

Metric Learning では、扱う距離指標や距離空間について異なる 2 つのアプローチがある。1 つは曲率が異なる一般的な空間や、一般化された測地線で距離を測る多様体上にデータをマッピングするアプローチ [37, 38] である。これはデータセットやそのドメインにおける各データ間の距離を正確に獲得することができると考えられる。しかしその一方で、データ間の距離を計算するコストが増大したり、データ不足などで変換や空間を近似しながら獲得することが困難であるといった課題がある。

もう 1 つのアプローチは、空間にはユークリッド空間を用いて、データやその特徴量同士のユークリッド距離が適切になるような変換を学習により獲得するアプローチである。距離を求める計算コストを抑えることで高速な推論が可能であるなどの観点から、ユークリッド距離に基づいて特徴量抽出モデルを学習させる研究が多く提案されている。本研究でもこちらのアプローチにより、画像データ同士がクラスに応じて所望の距離を持つような特徴量空間に変換する Deep Neural Network の学習を行う。

2.3　Deep Metric Learning

2.3.1　画像認識における Deep Metric Learning

近年の Deep Learning の発達により、CNN などのモデルを画像からの特徴量抽出器として用いることで、画像を数十から数百次元のベクトルに変換して高精度な Metric Learning を

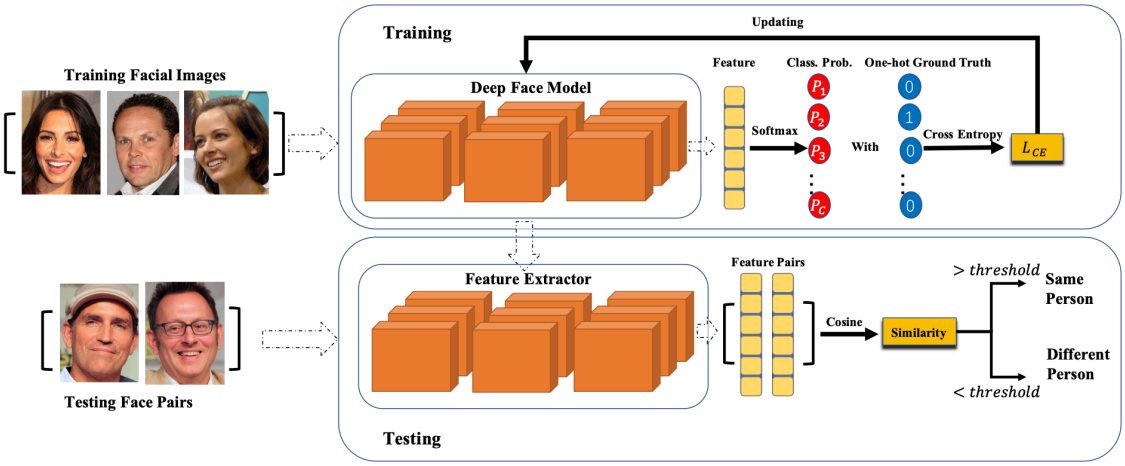


図 2.5　顔認識を例とする Deep Metric Learning[3]。

行うことが可能となっている。つまり画像を特徴量ベクトルの次元数で張られた空間中の点とみなしている。特に Deep Learning を用いた Metric Leaning を Deep Metric Learning と呼ぶ。例として、特徴量抽出器として Deep Learning モデルを用いた Deep Metric Learning の模式図を図 2.5 に示す。ここでは顔認証のタスクを例にとっているため Deep Face Model と表記している。

この特徴量空間において、特定のデータセットなどに対してクラスラベルに基づいた位置関係を獲得するために設計した損失関数が多く提案されている。具体的には、異なるクラス同士は距離をとってマッピングされる、各クラスは分散が小さくマッピングされるなどの制約条件を損失関数として定式化し、その値を用いて逆誤差伝播法を行うことで所望の特徴量を獲得する抽出器として Deep Neural Network を学習する。

2.3.2　損失関数

特徴量ベクトル間の距離に基づく損失関数を設計し、Deep Neural Network をはじめとする特徴量抽出モデルを逆誤差伝播法で学習する手法が多く提案されている。Contrastive Loss[17] は 2 つのデータを同一の特徴抽出ネットワークに入力し、出力された特徴量ベクトル間のユークリッド距離に基づいて損失関数を計算する。

$$L_{contrastive} = yd + (1 - y)[\alpha - d]_+$$

ここで、d は 2 点間の距離、y は 2 つのデータが同じクラスならば 1、異なるクラスならば 0 をとるラベルを表す。また $[]_+$ は $\max(x, 0)$ を表す。同一クラスのデータ同士は近づき、異

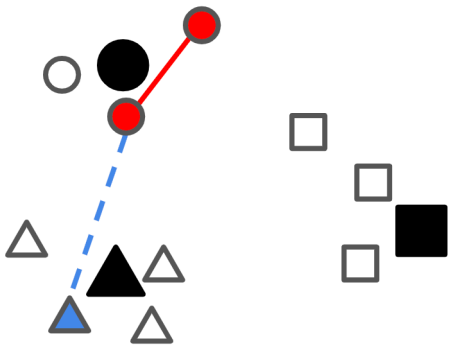


図 2.6　Triplet Loss。

なるクラスであれば一定のマージン α をもって離れるようにネットワークを学習する。この損失関数の課題として、同じデータ同士でもラベルによって近づけるか離すかが変わってしまう点が挙げられる。例として 2 人の男性の顔写真であれば、「男性」というラベルに基づくならば近づける必要があるが、異なる人物として識別するならば離す必要がある。

Triplet Loss[16] はこれを改良し、3 つのデータ間について距離を考える。あるデータ (anchor) に対し同じクラスのデータ (positive) と異なるクラスのデータ (negative) の 3 データを 1 セットとし、anchor と positive の距離にマージンを加えた大きさが anchor と negative との距離よりも大きくなるような損失関数である。

$$L_{triplet} = [d_p - d_n + \alpha]_+$$

ここで d_p は anchor と positive の距離、d_n は anchor と negative の距離、α はマージンを表す。2 次元平面における Triplet Loss の模式図を図 2.6 に示す。ここで、赤は anchor と positive、青は negative、黒がクラスの代表点や中心点として用いられる proxy、銭の太さが損失関数の大きさをそれぞれ表す。Contrastive Loss では同じデータセットでもタスクによって近づけたいのか遠ざけたいのかを考慮する必要があったが、この手法は 2 クラスを必ず同時に考慮して相対的な距離空間を得られる点で優れている。

2.3.3　発展的な Deep Metric Learning

前節の Contrastive Loss や Triplet Loss について、考えうるデータの組合せ数が膨大であり、学習に有効なペアが見つけづらいために収束が極端に遅くなってしまう問題点が指摘されている。これらの損失関数の拡張として、一度に考慮するデータクラス数を拡張した N-pair Loss[15] や Lifted Structure Loss[39] などが提案されている。これらは 1 つの

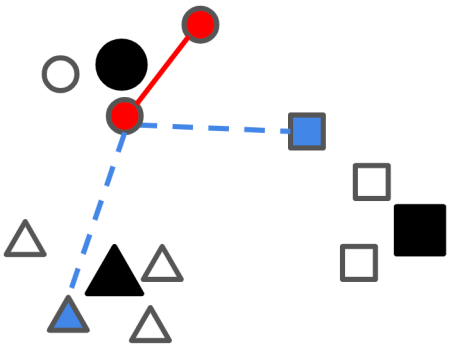


図 2.7　N-pair Loss。

anchor に対して 1 つの positive を近づけながら、複数の negative を離すような損失関数である。図 2.7 に N-pair Loss の模式図を示す。これらの手法は、学習用データセットのクラスを一度に多く考慮して学習を行うことができる。しかし一度の学習で各クラスから 1 つずつデータを用意して学習を繰り返しているに過ぎず、データセット全体の構造を捉えきるには不十分である。この問題点を解決するのが次章で説明する proxy を用いた学習方法であり、収束が早く高い評価スコアを与えることが知られる。

またこの学習に用いるデータの組み合わせ数が膨大である問題に対しては、モデルの学習に効果的なデータの組み合わせを探す hard sample mining といったアプローチが提案されている [18, 19, 40]。ある anchor に対して離れた positive sample や近くの negative sample が損失関数の値を大きくし学習を進める点に注目し、ミニバッチ内で学習に有効なペアを選ぶ手法 [18, 19] や、anchor に対して hard sample を生成する手法などが提案されている。すなわち現状のモデルにとってクラスの決定境界付近に位置する難しいデータを探す手法であると考えることができる。最新の研究 [41] では、学習の前半では easy sample を中心に学習を進め、後半では hard sample でより学習を進めることにより効率的に学習を行う手法も提案されている。

さらにこの easy/hard の考え方を応用し、特徴量空間中で学習に有効な hard negative を計算により生成してデータのペアを作ることで hard sample mining なしに学習を進める手法 [4] がある。図 2.8 にこの手法を 2 次元平面で可視化した概念図を示す。まず同一クラスの 2 データを変換した特徴量ベクトル間でいくつかの内分点を求める。これを他クラスで同様に算出し、位置関係的にもっとも損失関数が大きくなる 2 点をもとに損失関数を計算する。

これらの方法は学習中のモデルにとって識別の難しいデータの組を作成することで効率

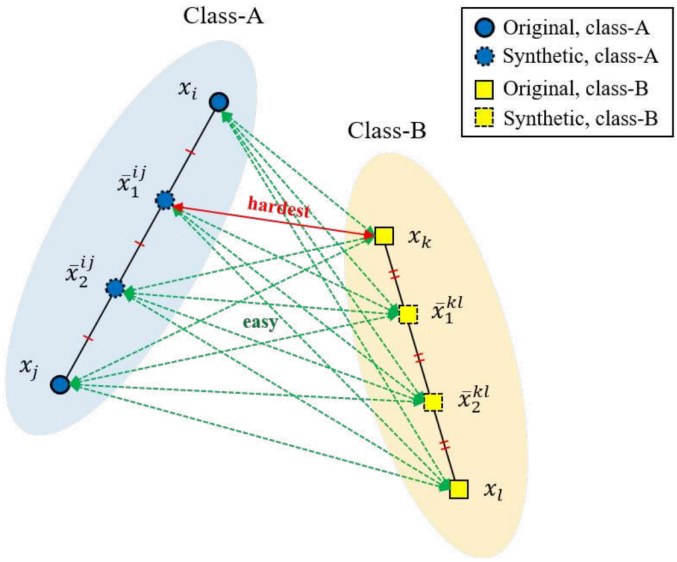


図 2.8　特徴量空間で学習に有効な位置を計算し、データ点として生成する手法 [4]。

的にモデルの学習を行うことができるが、探索を行う際に学習に直接は不要な距離の計算が必要となるため追加の計算コストがかかるといった課題がある。

第3章　分類型ネットワークで行うMetric Learningとスケーリングパラメタ

3.1　代表点とクラス分類で行うMetric Learning

3.1.1　proxy: 各クラスの代表点

学習に用いるデータの組み合わせ数が膨大で収束が遅いDeep Metric Learningの課題に対して、proxyとよばれる各クラスの代表点のみに対して既存の損失関数を計算する手法が提案されている[20, 10, 9, 42, 5]。所望の特徴量空間を獲得するために、個別のデータでなく代表点のみで位置関係を考えることができれば、学習の収束は格段に速くなる。表3.1に、既存研究で提案されている種々の損失関数と考えられるデータの組み合わせ総数を示す。ここで、Mは学習用データの総数、Cはクラス数、Bはバッチサイズ、Uは1クラスあたりのproxy数を表す。2つのデータ間の距離[17]や3つのデータ組であるtriplet[16]間の距離をラベルに基づいて近づけたり離したりするような損失関数では、学習用の全データ数Mの二乗や三乗が組み合わせの総数となる。このうち特徴量空間において境界付近に位置するhard positiveやhard negativeは数が少なく、有効なデータの組を見つけられずに学習の収束が遅くなってしまう。これらに対しバッチサイズを大きく取り学習に有効なnegative sampleを探してtripletを作るFaceNet[21]や、anchorとなるベクトルに対して学習に有効なpositiveとnegativeを探索するSmart Mining[18]など、効率的に学習を進める手法が提案されている。しかし、本研究で使用したデータセットでもデータの総数は10^5オーダーであり、学習を進めるのが困難であることに変わりはない。

一方で、データセットの各クラスは最小で10^2程度のオーダーである。MegaFace[43]など顔画像の大規模なデータセット等で約600,000のクラス(ID)が含まれるが、本研究で検証する代表的なMetric Learningの画像データセットはクラス数が200程度である。これら各クラスを表す代表点を特徴量空間中に保持し、代表点同士でペアやtripletを作成して学習を行うことで、速く学習を進めることができることが知られる[20]。図3.1にProxy-NCAの模式図を示す。こうすることでデータの組み合わせ数をデータ数Mについてのオーダー

表 3.1　損失関数に対する計算量のオーダ比較。

タイプ	Loss	データの組み合わせ総数
pair-based	Contrastive	$O(M^2)$
	Triplet(Semi-Hard Mining)[21]	$O(M^3/B^2)$
	Triplet(Smart Mining)[18]	$O(M^2)$
	N-pair[15]	$O(M^3)$
	Lifted Structure[39]	$O(M^2)$
proxy-based	Proxy Anchor[10]	$O(MC)$
	Proxy-NCA[20]	$O(MC)$
	SoftTriple[9]	$O(MCU^2)$

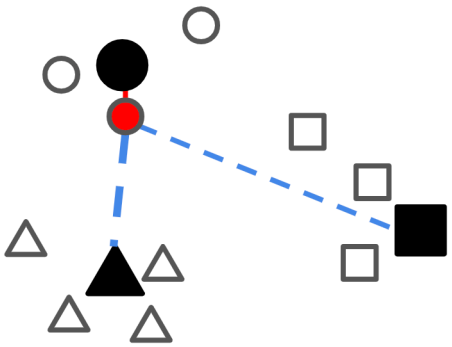


図 3.1　Proxy-NCA。

からクラス数 C についてのオーダーに落とすことができ、学習を速く収束させることができる [10]。なお proxy ベクトルは学習の開始時に各次元の要素を正規分布などに基づく簡易な手法で初期化され、ネットワークと同時に学習され更新されていくことが多い。

3.1.2　Cross Entropy Loss を応用した Metric Learning

近年顔認証の分野を中心に、全結合層を接続したクラス分類型ネットワークで Metric Learning を行う手法が高い識別能力を獲得できることが知られている [11, 25, 26]。CNN などによって画像から得られる特徴量に対して、学習データのクラス数に対応する全結合層を接続し、クラス分類として学習する。つまり、距離の計算や学習に有効なペアを探す必要がなくなり、学習とその収束が速くなる。Cross Entropy Loss を応用した Metric Learning

で、代表的な損失関数である ArcFace[11] は

$$L_{ArcFace} = -\frac{1}{N}\sum_{i=1}^{N}\log\frac{e^{f_{i,y_i}}}{e^{f_{i,y_i}} + \sum_{k\neq y_i} e^{f_{i,k}}}$$

と表すことができる。ここで、$f_{a,b}$ は空間中の点 a と点 b の類似度、N はミニバッチ内のデータ数、y_i はデータ i が属するクラスのラベルである。

proxy を用いた手法では逐次全データベクトルの座標を平均して中心点を算出するのではなく、要素毎に初期化を行った同次元数のベクトルをクラス数分だけ保持する。このベクトルにも学習率を設定し、損失関数の値を用いてネットワーク部と同様に学習を進めていく実装がされている。一方 ArcFace のような損失関数は、特徴量次元層の後段に全結合層を接続して重みを保持し、L2 正規化を行って内積計算が行われる。つまり、proxy を用いた Metric Learning と、ArcFace をはじめとするクラス分類型ネットワークによる学習法は同一の考え方であると解釈することができる。

3.2　スケーリングパラメタ

3.2.1　cosine 類似度でクラス分類を行う問題点

データの特徴量ベクトルやクラスの代表点ベクトルを L2 正規化し、特徴量空間を次元内の単位円上に制約することで、良い特徴量空間を獲得できることが知られている [22, 23, 44]。この時に 2 点のベクトルノルムが 1 であることから、内積を計算しなす角 θ を用いた $\cos\theta$ を類似度の指標として用いることができる。cosine 類似度を用いると、上述の損失関数 ArcFace は次のように表される。

$$L_{ArcFace} = -\frac{1}{N}\sum_{i=1}^{N}\log\frac{\exp(s\cos(\theta_{y_i}+m))}{\exp(s\cos(\theta_{y_i}+m)) + \sum_k \exp(s\cos\theta_k)}$$

ここで、θ_{y_i} は入力データの特徴量ベクトルとそのクラスの重みベクトルとの角度、θ_k は入力データの特徴量ベクトルと他クラスの重みベクトルとの角度、s はスケーリングパラメタ、m はマージンをそれぞれ表す。マージンは、各データについて自身が属するクラスベクトルとの距離にのみ追加の制約を設ける役割があり、各クラス内の分散を小さくする [45]。上述のネットワークアーキテクチャにおいて最終的に損失関数を計算するにあたり、$\cos\theta$ の値のみが Softmax 関数へ入力されることになる。

しかし、これにより Softmax 関数の入力値は $-1 \leq \cos\theta \leq 1$ となる。これでは、正解となるクラスを高い確度で判別できていても、すなわち該当するクラスの cosine 類似度が最

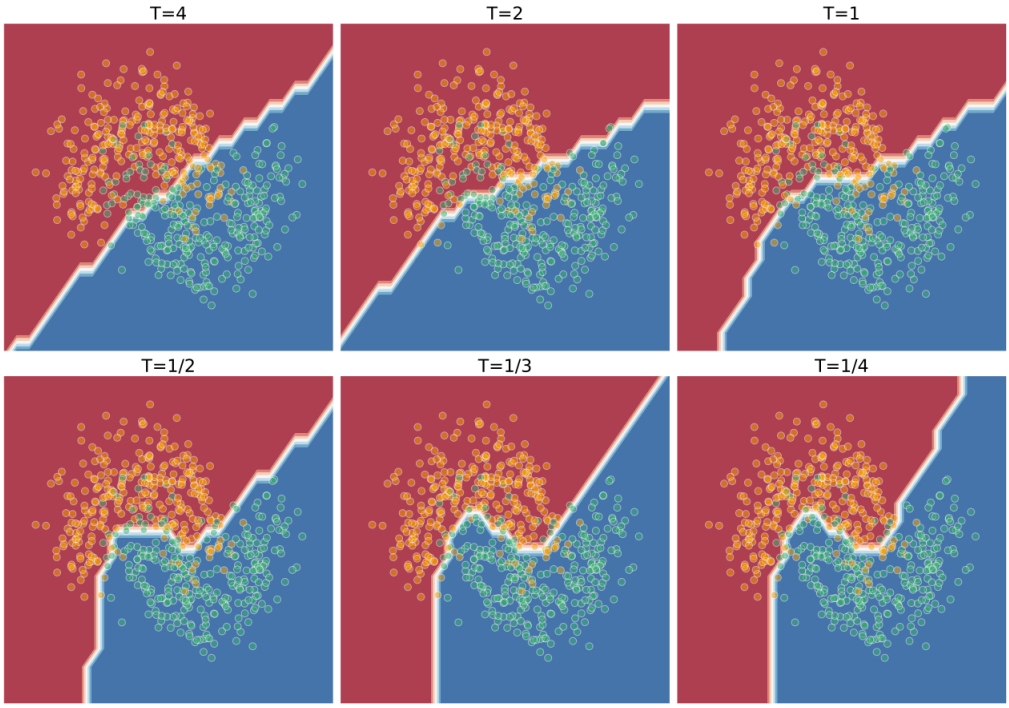


図 3.2　温度パラメタによって変化する 2 クラス間の決定境界 [5]。

大である 1 に近くても、他クラスの類似度と差が大きく開かない。その結果として損失関数の値が大きく変わらないため、学習がうまく進まない。そこで cosine 類似度のような値域の幅が狭い入力を Softmax 関数に適用する際、以下に説明するスケーリングパラメタが導入される。

3.2.2　新しいパラメタの導入

DNN の知識蒸留を提案した研究 [27] では、学習済みモデルが出力する予測値全体をより小さなモデルの学習に使用する目的で温度パラメタ T が導入された。これは Softmax 関数への入力ベクトルに T^{-1} をかけることで、教師モデルの出力と生徒モデルの出力の差を強調して学習を大きく進めるはたらきを持っている。クラス分類のために cosine 類似度を用いて Metric Learning を行う手法においても同様のパラメタが用いられ [46]、スケーリングパラメタやスケーリングファクタと呼び、Softmax 関数への入力を定数倍する係数として導入される。図 3.2 に、2 次元のデータを温度パラメタつき Softmax 関数で分類した時に決定境界が変化する様子を示す。T が小さくなり Softmax 関数への入力をスケーリングするパラメタとしては大きい係数となることで、クラス間の決定境界が複雑になることが

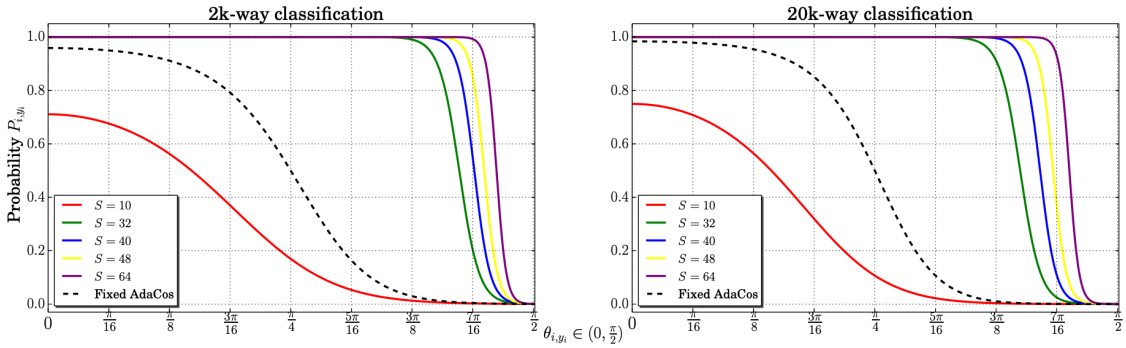


図 3.3　クラスベクトルとデータベクトルがなす角度および両者が同一クラスである確率について、スケーリングパラメタの値を変化させた時の閾値の変化 [6]。

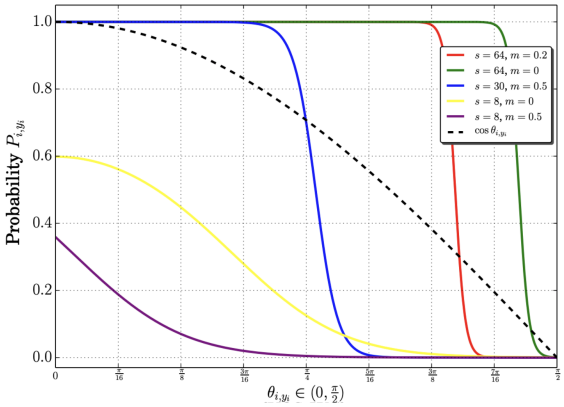


図 3.4　スケーリングパラメタとマージンによる確率の閾値の変化 [3]。

示されている [5]。つまり小さすぎるスケーリングパラメタでは識別能力が十分でなく、また大きすぎるスケーリングパラメタでは学習データに過学習してしまう。どちらにしても未知のクラスであるテストデータに対する汎化性能が下がってしまうと考えられる。

Metric Learning の Recall@K スコアはこのスケーリングパラメタに依存して大きく変動することが知られており [10]、特定のスケーリングパラメタで高いスコアを記録する手法でも、スケーリングパラメタを変えるとスコアが伸びない。このようなハイパーパラメタは一般に経験的、あるいはグリッドサーチやランダムサーチ [47] をはじめとする探索手法や、ベイズ最適化 [48] などで最適な値を決定する。

図 3.3 に、スケーリングパラメタを変化させた時の、ベクトル間の角度による同一クラス確率を表したグラフ [6] を示す。また図 3.4 に、マージンとともに変化させた時の確率を表したグラフを示す。

図 3.3 左がクラス数 2000、右がクラス数 20000 の時をそれぞれ表している。図中の Fixed AdaCos は、後述するクラス数に応じて最適な定数スケーリングパラメタを算出する手法 [6] である。スケーリングパラメタが大きい時、クラスベクトルとデータの特徴量ベクトル間の角度が大きく開いていてもそのクラスに属していると判定され、損失関数の値が大きくなる。すなわち、特徴量空間においてクラス同士の決定境界付近に位置しているような hard positive なデータでも学習が進みにくい。逆にスケーリングパラメタが小さい時にはクラスベクトルとデータの特徴量ベクトル間の角度がほとんど 0 に近くてもそのクラスに属している確率が低いと判定される。この時、学習を進めるのに有効でないような easy positive に対しても損失関数の値が大きくなってしまい、学習が収束しない。以上の既存研究による報告と考察から、スケーリングパラメタにはネットワークの学習において適切な値の範囲があると考えられる。

3.2.3　AdaCos

このスケーリングパラメタについて、用いるデータセットのクラス数をもとに最適な値が決定可能であるとする研究がある [6]。この研究では、クラス数が C のときの最適なスケーリングパラメタは

$$s_{init} = 2\log(C-1)$$

で与えられると主張している。また、$B_i^{(t)}$ を各データについて自身が属さないクラスとの類似度の和とし、その平均値 $B_{average}^{(t)}$ を、ミニバッチに含まれるクラスの集合を M を用いて

$$\begin{aligned} B_{average}^{(t)} &= \frac{1}{N}\sum_{i\in M} B_i^{(t)} \\ &= \frac{1}{N}\sum_{i\in M}\sum_{k\neq y_i} \exp(s^{(t-1)}\cos\theta_{i,k}) \end{aligned}$$

と表すとすると、学習中、t 回目の epoch において最適なスケーリングパラメタは

$$s^{(t)} = \frac{\log B_{average}^{(t)}}{\cos(\min(\frac{\pi}{4}, \theta_{median}^{(t)}))}$$

で与えられると主張している。ここで、θ_{median}^{t} は各ミニバッチにおける角度分布の中央値を表す。

この手法の課題として、各 epoch の各ミニバッチにおいてデータや代表点同士の角度を全て算出し平均値や中央値を求める必要があるために学習コストが増大してしまう点が挙

げられる。またこの研究は非常にクラス数の多い顔画像データセットを用いて有効性を示しているが、この手法で算出されるスケーリングパラメタの理論値はよりクラス数が少ない画像データセットも使用する Deep Metric Learning の研究において、経験的に用いられている値とは大きく異なっている。proxy を用いて全データをもとに損失関数を計算する手法を提案している既存研究 [10] でも、損失関数の提案に加えてスケーリングパラメタが評価スコアにどう影響するかを報告しており、比較したところ学習がよりうまく進み評価スコアが高くなるようなスケーリングパラメタは、AdaCos で求められる値よりも大きい範囲に存在していることが分かった。

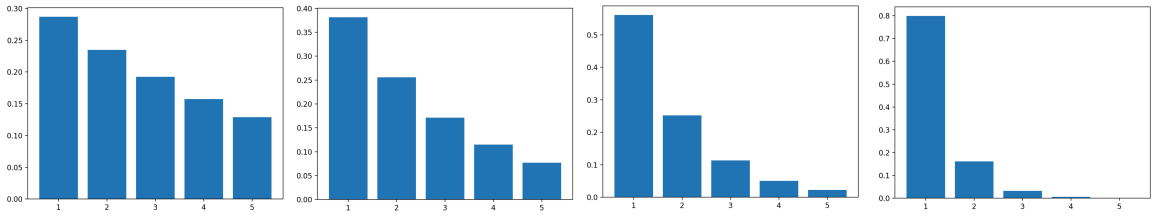
また、スケーリングパラメタを学習の途中で小さい切り替えることで評価スコアが向上する手法も提案されている [28]。しかし、学習の初期に値を切り替えているため、データセットをはじめとする諸条件が変化した際に学習が安定するとは限らない。また用いるスケーリングパラメタの初期値や最終値、減少のさせ方など最適な切り替え方に関する検討は行われておらず、実験的に得た初期値と最終値が有効であることのみが確認されている。

3.2.4　提案手法

図 3.5 に、同一の入力に異なるスケーリングパラメタを乗じた際の Softmax 関数の出力を示す。この図に示すように、スケーリングパラメタには Softmax 関数をより max 関数に近づける役割がある。つまり、Softmax 関数への入力間で差が小さい場合、全ての値を定数倍することで差を強調している。

cosine 類似度を用いた Deep Metric Learning では、モデルの学習が進むにつれてクラス間のなす角度が大きくなっていく [6]。したがって特徴量ベクトルやその proxy 間の類似度の差が小さい学習初期には大きなスケーリングパラメタを用い、クラス間の差異を強調することが望ましい。一方、学習が進むにつれてクラス間の距離は離れていくので、スケーリングパラメタを小さくし、cosine 類似度自体の差がより大きくなるように学習するのが望ましい。つまり学習の進展に伴って同一クラス内の cosine 類似度が大きくなり、同時にクラス間の cosine 類似度が小さくなるにつれて、スケーリングパラメタによる強調を弱めるのが学習に有効であると考えられる。

これらの考察を踏まえ、通常の学習が終了している Deep Metric Learning モデルに対して、スケーリングパラメタを小さくしながら追加学習を行う手法を提案する。具体的には、スケーリングパラメタを線形に減少させながら、学習に必要とした epoch 数と同じだけの epoch 数で追加の学習を行う。この有用性を確認するため、次章に述べる一連の実験を行っ



(a)　(b)　(c)　(d)

図 3.5　同一のベクトルに異なるスケーリングパラメタ s を乗じて Softmax 関数に入力した時の出力の変化。(a)$s = 1$、(b)$s = 2$、(c)$s = 5$、(d)$s = 10$

た。実験では、最適なスケーリングパラメタを算出する既存手法 [6] がクラス数の少ないデータセットにも拡張可能であるかを確認した後に、通常学習後にスケーリングパラメタを減少させて追加の学習を行う手法を提案し、Recall@1 スコアの変化により検証する。

さらに 5 章では、4 章の実験結果と考察からさらにスケーリングパラメタが学習に与える影響についていくつかの実験を行った。具体的には、学習中におけるクラスの代表点間の角度分布の変化、学習初期からスケーリングパラメタを線形に減少させる手法、追加学習の際にスケーリングパラメタを 2 次関数的に減少させる手法である。

第4章　減少するスケーリングパラメタを導入した追加学習

4.1　実験条件

4.1.1　データセット

データセットには、画像認識の分野において Metric Learning の性能評価に使用される CUB-200-2011 データセット [7] と Cars-196 データセット [8] を用いた。それぞれ図 4.1 と図 4.2 にデータセット中に含まれる画像の一例を示す。

前者は 200 クラスにラベル付けされた、各クラス 20 枚ほどの縦横それぞれ約 300 ピクセル程度の鳥の画像 11,788 枚からなる。後者は 196 クラスにラベル付けされた、各クラス 70〜100 枚ほどの縦横約 200〜700 ピクセル程度の車の画像 16,185 枚からなる。

4.1.2　評価指標

Metric Learning においては、獲得した変換や特徴量空間の性能評価に Recall@K スコアという指標が用いられる。本研究でも既存の関連研究と同じく、このスコアで K=1 とした Recall@1 を用いて評価実験を行っている。このスコアは獲得されたネットワークでテストデータを特徴量空間に投影し、各データ点において近傍 K 個の点に同一クラスが含まれている割合で計算される。例として Recall@2 であれば「Recall@1 すなわち最近傍のデータ点が同一クラスである割合」と「最近傍点は自身と異なるクラスだが 2 番目に近い点は同じクラスである割合」の合計を表す。

推論時に未知のクラスがあることを想定した画像認識タスクでは、Recall@K スコアは未知のクラスに対して計算することが多い。本研究でも、Recall@1 スコアは未知のクラスのみに対して計算した。具体的には、画像データセットをクラス数で 2 分割し、前半のクラスに属する画像でネットワークの学習を行ったのち、残り半分のクラスに属する新たな画像をネットワークに入力する。得られた特徴量空間内のテストデータ点について、最近傍

図 4.1　CUB-200-2011 データセット [7]。

図 4.2　Cars-196 データセット [8]。

のデータ点が同一クラスであるかどうかを全てのデータについて確認することでテストスコアを計算する。

4.1.3　損失関数

損失関数には、クラス分類型ネットワークで Deep Metric Learning を行う手法である SoftTriple Loss[9] を使用した。これは「特徴量空間において、各クラスを代表するような点は複数ある」との考察に基づいたもので、同じクラスに属する画像でも見た目が異なるものがあり、それらは特徴量空間ではいくつかの小さいクラスタに分かれて分布することが想定されている。例えば CUB-200-2011 データセットの同一クラス中にもオスとメスで色が違っていたり、木に止まっている画像と海面上を飛んでいる画像が含まれていたりなど明らかに見た目の異なる画像がある。この論文中ではこれらを「一つのクラスが複数の

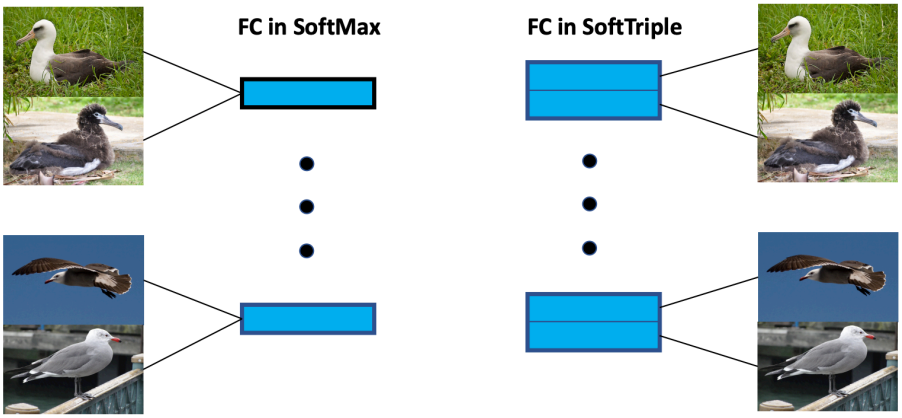


図 4.3　同一クラスで見た目が異なるデータの例 [9]。

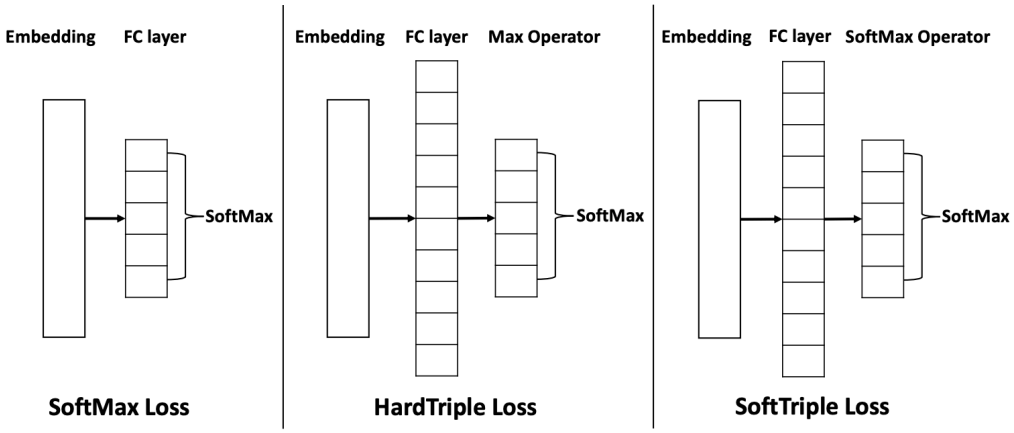


図 4.4　SoftTriple Loss の全結合層部 [9]。

代表点を持つ」ことで対応可能であるとしている。この様子を図 4.3 に示す。これは 2.2 節で述べた Cross Entropy Loss を用いた Metric Leanring の拡張にあたり、代表点の数 K が 1 の時に通常の分類型ネットワークと同じものとなる。この損失関数は畳み込みとプーリングを繰り返して得られた特徴量ベクトルを、クラス数と代表点の数の積だけ次元を持つ全結合層に接続することで実装される。この模式図を図 4.4 に示す。

図 4.4 それぞれの中央にある FC Layer は各クラスにつき複数個の代表点ベクトル (全結合層の重み) を持つ構造を表している。またこの損失関数は以下の式で表される。

$$L_{SoftTriple}(x_i) = -\log \frac{\exp(\lambda(S'_{i,y_i} - \delta))}{\exp(\lambda(S'_{i,y_i} - \delta)) + \sum_{j \neq y_i} \exp(\lambda S'_{i,j})}$$

ここで、λ はスケーリングパラメタ、S' は relaxed similarity、δ はクラス間の距離指標に持

たせるマージンを表す。relaxed similarity は図 4.4 右の SoftTriple Loss で計算される類似度である。FC layer は代表点数×クラス数の次元を持っており、各クラスについて代表点ごとに計算された類似度をまとめ、後段に接続された最終の全結合層にそれぞれのクラスに属する確率として渡す構造になっている。このまとめる操作を Max 関数で行うか Softmax 関数で行うかの違いで HardTriple と SoftTriple の二種類の損失関数がある。HardTriple Loss に含まれる Max 関数は非連続であるため、元の論文でも SoftTriple Loss を使用することを提案している。代表点との類似度全てを Softmax 関数でまとめているのが relaxed similarity であり、以下の式で表される。

$$S'_{i,c} = \sum_k \frac{\exp(\frac{1}{\gamma} x_i^T w_c^k)}{\sum_k \exp(\frac{1}{\gamma} x_i^T w_c^k)} x_i^T w_c^k$$

ここで、x_i は i 番目のデータの特徴量ベクトル、c はクラス、$\frac{1}{\gamma}$ はスケーリングパラメタを表す。なお、本研究ではこちらのスケーリングパラメタは定数を用いている。

またこの損失関数の最小化を考える際には、学習の過程で各クラスについて代表点同士を近づけマージしていくような制約項も含まれる。この項を加え、今回の学習における最小化の目的関数は以下の式で表される。

$$\text{minimize } \frac{1}{N} L_{SoftTriple} + \frac{\tau \sum_j^C R(w_j^1, ..., w_j^K)}{CK(K-1)}$$

ここで、N は学習用データの総数、τ は定数係数、C は学習用データのクラス数、$R(...)$ は j 番目のクラスについての代表点を近づける制約式、K は 1 クラスあたりに設ける代表点数の初期値を表す。具体的には $R(...)$ は以下の式で表される。

$$\begin{aligned} R(w_j^1, ..., w_j^K) &= \sum_t^K \sum_s^K ||w_j^s - w_j^t||_2 \\ &= \sqrt{2 - 2 w_j^{sT} w_j^t} \end{aligned}$$

なお、次節で説明する実験条件および各パラメタの値について、スケーリングパラメタを除いてこの論文 [9] に従って実験を行っている。

4.1.4　その他実験条件

実装には、Python で Deep Learning モデルの実装を行うパッケージツールである PyTorch[49] を用いた。実験に用いる backbone ネットワーク (画像から特徴量を抽出する CNN) には、

画像データセットである ImageNet[50] で事前学習済みの GoogleNet[31] アーキテクチャを使用した。先行研究 [6, 28] でも backbone にこのアーキテクチャを使用しており、本実験でも Batch Normalization[32] を行うモデル GoogleNet-v3 を採用した。このネットワークの最終層の次元をそれぞれのデータセットのクラス数の半分の値 (CUB-200-2011 データセットは 100、Cars-196 データセットは 98) に変更し、全体の半分のクラスを学習用データとして学習を行った。

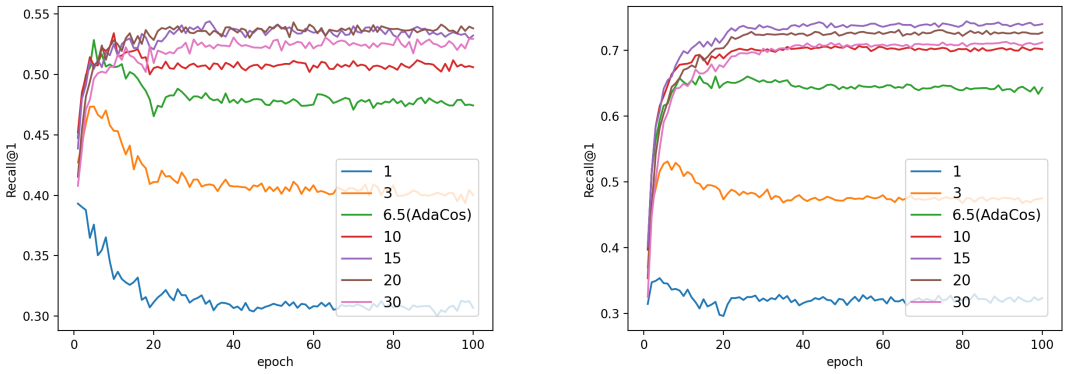
学習中には、データの水増しとして左右反転と random crop を適用した。左右反転は 50%の確率で行われ、random crop はデータ画像のランダムな場所を指定したピクセルサイズで切り取る処理である。テストデータには学習に使用していない残り半分のクラスを使用し、center crop のみを適用した。これは画像の中心を指定したピクセルサイズで切り取る処理である。学習と推論どちらにおいてもデータ画像は 224 ピクセル四方に crop した。backbone ネットワークと代表点ベクトルの学習率はそれぞれ 0.0001 と 0.01 とし、20 および 40 エポック目にそれぞれ学習率を 0.1 倍した。特徴量の次元数は 64、バッチサイズは 32、各クラスの代表点は初期値 10、クラス間のマージン δ は 0.01、各代表点との類似度計算を行う際のスケーリングパラメタ γ を 0.1(つまり入力は 10 倍される)、同一クラスの各代表点を近づける制約項の係数 τ は 0.2 とした。また、確率的勾配降下法の最適化アルゴリズムには、ネットワークと代表点ともに Adam[51] を使用した。

4.2　実験

まず、定数スケーリングパラメタの値を変えてネットワークを 50epoch 学習させる実験を行った。次に、前の実験で評価スコアの高かったスケーリングパラメタで 50epoch 学習させてから、スケーリングパラメタを線形で減少させて追加で 50epoch 学習させる実験を行った。最後に、線形に減少させる手法と追加学習開始時にスケーリングパラメタを小さな値に切り替える手法との比較を行った。

4.2.1　実験 1: 最適な定数スケーリングパラメタ

スケーリングパラメタに定数として 1, 3, 6.5, 10, 15, 20, 30 を設定して学習を行った。なお、通常の学習は 50epoch であるが、これ以降の実験で追加の学習をさらに 50epoch 行う際の比較のため、計 100epoch の学習を行っている。学習中の Recall@1 スコアの変化を図 4.5 に、学習終了時の Recall@1 スコアを表 4.1 にそれぞれ示す。6.5 が既存手法 AdaCos[6] の計算式により求められる最適とされるスケーリングパラメタである。



(a)　(b)

図 4.5　定数スケーリングパラメタで学習した時の Recall@1 スコアの変化。(a)CUB-200-2011、(b)Cars-196

表 4.1　実験 1 における学習終了時の Recall@1 スコア。

データセット	スケーリングパラメタ	Recall@1
CUB-200-2011	1	0.3068
	3	0.4001
	6.5	0.4743
	10	0.5060
	15	0.5323
	20	**0.5351**
	30	0.5295
Cars-196	1	0.3232
	3	0.4750
	6.5	0.6433
	10	0.7019
	15	**0.7401**
	20	0.7273
	30	0.7121

結果より、スケーリングパラメタは Deep Metric Learning の学習に大きく寄与していることがわかる。10 以下の小さな値ではテストスコアが上がっておらず、15 から 30 が学習をうまく進めるために十分な値であることがわかる。また、20 から 30 にかけてスコアが低くなっている。この結果より、スケーリングパラメタの値が十分に大きくないと学習が

表 4.2　Cars-196 データセットを用いて Proxy Anchor Loss[10] で学習を行った時の、スケーリングパラメタとマージンによる評価スコアの違い。

スケーリングパラメタ	マージン				
	0	0.1	0.2	0.3	0.4
4	64.46	65.56	66.68	68.24	69.09
8	76.40	77.94	77.73	79.15	79.14
16	84.11	83.66	83.51	83.58	84.52
32	86.29	86.10	85.66	85.71	85.13
64	83.66	86.67	86.26	85.51	86.35

うまく進まないことが分かる。AdaCos[6] に基づいて定めた最適なスケーリングパラメタは 6.5 であり、Recall@1 スコアは CUB と Cars でそれぞれ 0.4743 と 0.6433 に収束した。しかし図 4.1 と表 4.1 からも分かる通り、より大きなスケーリングパラメタを用いた方が良いスコアが得られている。今回の実験ではスケーリングパラメタがそれぞれ 20 と 15 の時に最良となり、Recall@1 は 0.5381 と 0.7401 となった。また逆に大きな値でもテストスコアが下がっていることを確認した。これは 3 章で考察したように、学習用データに含まれるクラス間の決定境界に過学習してしまった結果テスト用データに対する汎化性能が下がっているためであると考えられる。

また、同じように cosine 類似度と代表点を用いる Proxy Anchor Loss を提案している論文 [10] でも、16 程度の値から十分に高い評価スコアが得られるとしている。表 4.2 に、この論文中で検証されているスケーリングパラメタとマージンの影響を示す。マージンの値はスコアを高めるためには調整が必要不可欠であるが、学習の進行に影響する大きな違いが現れるのはスケーリングパラメタの値であることが主張されており、表 4.2 の結果からもスケーリングパラメタの重要性がわかる。この Proxy Anchor Loss も本実験で用いた SoftTriple Loss も cosine 類似度と代表点を用いる損失関数であり、同様の結果が得られたと言える。

また図 4.5 より、スケーリングパラメタが小さい時、学習の進展と共に Recall@1 スコアが上昇した後、減少する傾向がみられる。この原因について検証するために追加の実験を行った。具体的には、proxy 間の角度が学習中どう変化しているかを見ることで、過学習を起こしている可能性について検討する。この詳細については次章で述べる。

表 4.3 実験 2 における通常学習と追加学習終了時の分類精度と Recall@1 スコア。

	最終値	分類精度		Recall@1		Recall@1 の変化
		50epoch	100epoch	50epoch	100epoch	
CUB-200-2011	20	0.9158	0.9150	0.5361	0.5351	-
	10	0.9155	0.9177	0.5331	0.5432	+0.0101
	5	0.9148	0.9119	0.5366	**0.5533**	+0.0167
	2.5	0.9112	0.9043	0.5283	0.5453	**+0.0170**
Cars-196	20	0.9100	0.9190	0.7357	0.7340	-
	10	0.9240	0.9187	0.7163	0.7302	+0.0139
	5	0.9230	0.9136	0.7266	0.7465	**+0.0199**
	2.5	0.9107	0.9033	0.7290	**0.7471**	0.0181

4.2.2　実験 2: 線形減少するスケーリングパラメタを用いた追加学習

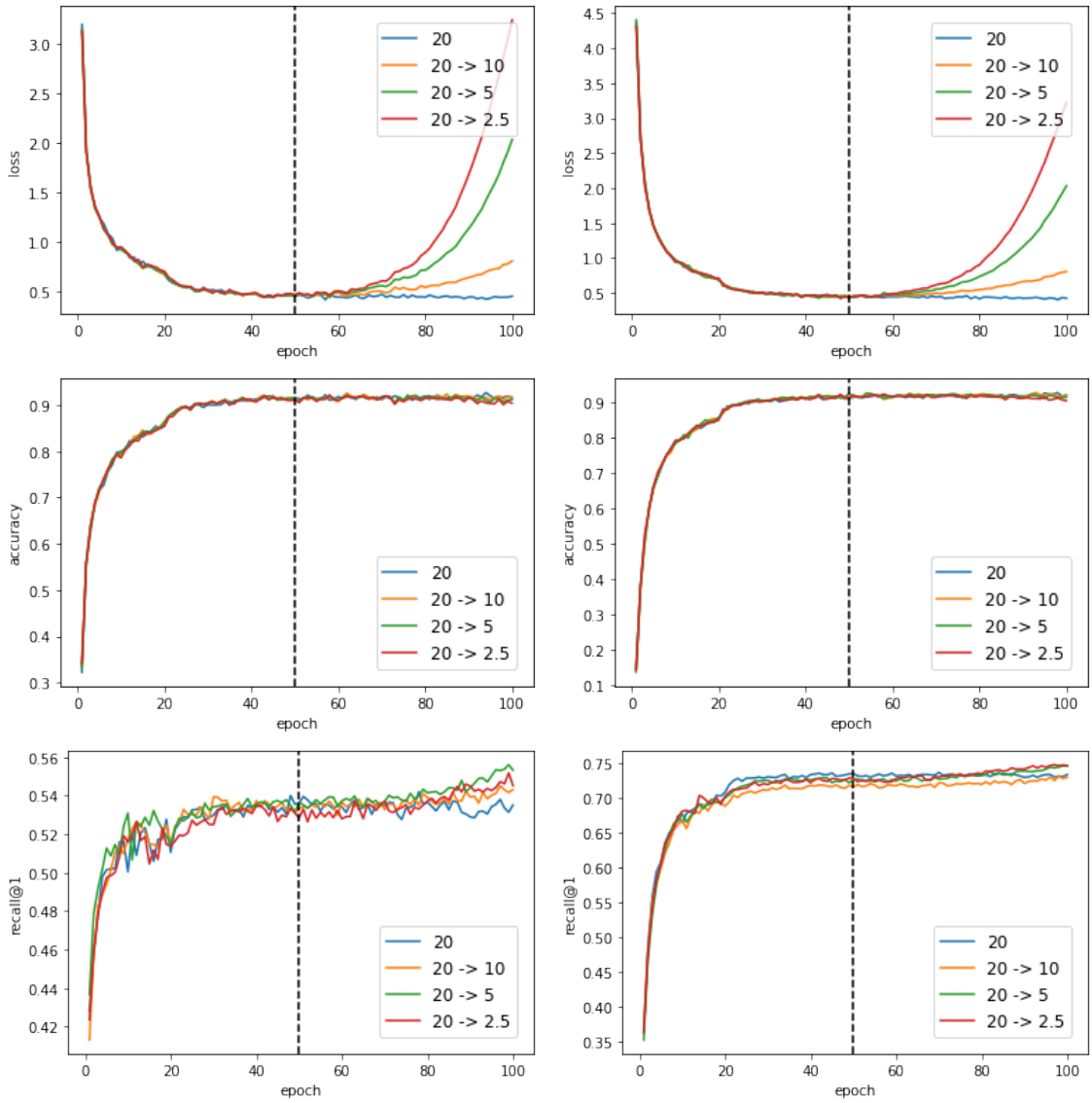
通常学習を定数スケーリングパラメタで 50epoch 進め、続いてスケーリングパラメタを線形に減少させながら 50epoch の追加学習を行った。この定数には通常の学習を良い状態で行うため、実験 1 の結果より 20 を設定した。この実験中の損失関数の値、分類精度、テストデータの Recall@1 を図 4.6 に示す。また、通常学習と追加学習の終了時について、分類精度と Recall@1 を表 4.3 に示す。

CUB-200-2011 と Cars-196 両データセットで、通常学習の終了時に対して追加学習を行うことで全ての最終値について Recall@1 スコアが改善した。通常学習の終了時にスコアにばらつきが見られるが、これは実装の都合上全ての条件設定について通常学習から行っているためである。

この実験で用いた最終値は、実験 1 において図に 4.5 に示すように、定数で学習を行った際にはうまく学習が進まなかった範囲である。最初は十分に大きい値で学習し、学習が進むとスケーリングパラメタによる強調を弱めることでモデルの学習がより進んでいることが確認された。以上より、提案手法である学習初期と後半でスケーリングパラメタを切り替えるアプローチが有効であることが分かった。

4.2.3　実験 3: 既存手法との比較

同様に小さなスケーリングパラメタで学習を行う手法として、追加学習の開始時に小さな値に切り替える手法についても同様に実験を行った。図 4.7 に学習中の Recall@1 スコア



(a)　(b)

図 4.6　スケーリングパラメタを線形に減少させながら追加学習を行った実験における損失関数の値、分類精度、Recall@1 スコアの変化。(a)CUB-200-2011、(b)Cars-196

の変化を、表 4.4 と 4.5 に、通常学習終了時、追加学習終了時 (100epoch 目と 150epoch 目) における分類精度と Recall@1 スコアをそれぞれ示す。なお実験 2 において損失関数が追加学習の進行に従って増加している点に着目し、50epoch の追加学習の後にさらに 50epoch の学習を行った。この最終の 50epoch についてはパラメタは変更していない。

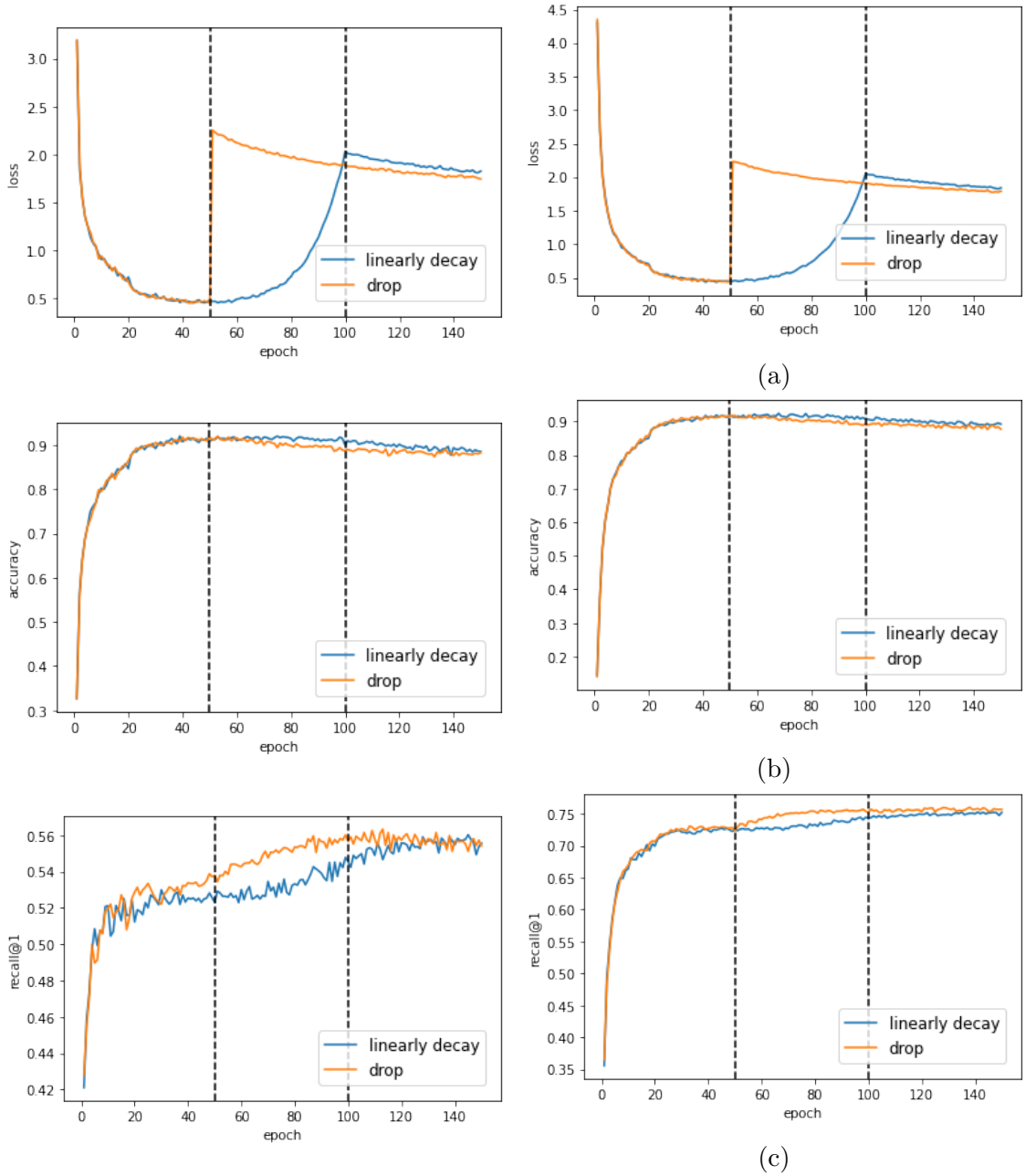


図 4.7　異なる手法でスケーリングパラメタを減少させた実験における損失関数の値、分類精度、Recall@1 スコアの変化。(a)CUB-200-2011、(b)Cars-196

スケーリングパラメタを線形減少と切り替えの両手法で変化させることで、ともに追加学習により評価スコアが向上することを確認した。表 4.4 と 4.5 より、スケーリングパラメ

表 4.4　実験 3 における通常学習終了時と追加学習終了時の分類精度と Recall@1 スコア (CUB-200-2011)。

評価値	スケーリングパラメタ	50epoch	100epoch	150epoch	50epoch からの変化
分類精度	線形減少	0.9080	0.9003	0.8855	-0.0225
	切り替え	0.9180	0.8871	+0.8823	-0.0357
Recall@1	線形減少	0.5230	0.5454	0.5555	+0.0325
	切り替え	0.5377	0.5566	0.5541	+0.0164

表 4.5　実験 3 における通常学習終了時と追加学習終了時の分類精度と Recall@1 スコア (Cars-196)。

評価値	スケーリングパラメタ	50epoch	100epoch	150epoch	50epoch からの変化
分類精度	線形減少	0.9166	0.9072	0.8922	-0.0244
	切り替え	0.9157	0.8932	+0.8773	-0.0384
Recall@1	線形減少	0.7251	0.7427	0.7522	+0.0271
	切り替え	0.7239	0.7519	0.7569	+0.0330

タを切り替える手法の方が評価スコアの向上が見られた。CUB-200-2011 データセットでは線形減少で 0.0167 ポイント、切り替えで 0.0208 ポイント、Cars-196 データセットでは線形減少で 0.0199 ポイント、切り替えで 0.0293 ポイントのスコア向上となっていた。

小さな値に切り替える手法では損失関数の値が急激に増加し、追加学習が進むに従って少しずつ下がっていた。一方線形に減少させる手法では、スケーリングパラメタが小さな値になるに従って損失関数の値が大きくなっていた。小さな値に切り替える手法が 100epoch 時点では分類精度が低く評価スコアは高いが、150epoch 終了時には逆転している。Metric Learning は既知のクラスについても分類精度を落とさずに未知のクラスの識別能力を獲得することが目的とされるため、トレードオフについてはタスク毎に検討する必要があると言える。

次章では、これらの結果と考察からさらにスケーリングパラメタの学習に与える影響について実験を行い検証する。

第5章　スケーリングパラメタが学習に与える影響

5.1　学習中におけるクラス代表点間の角度の変化

図 5.1 に、初期化された状態での代表点間の cosine 類似度の分布を示す。代表点ベクトルの初期化は先行研究と同様に Xaviar の初期化 (Glorot の初期化とも)[52] を用いて行っており、初期化時には代表点同士はおよそ 75 から 105° の角度で分布している。これは特徴量空間が 64 次元と高次元であり、ベクトルの各要素を正規分布などに基づいて初期化すると極めて高い確率でなす角度が 90 度に近くなるためであると考えられる。

続いて CUB-200-2011 データセットを用いて、スケーリングパラメタ s と特徴量次元数 d を変えた時に代表点同士の角度がどのように異なるか確認した。CUB-200-2011 データセットは全 200 クラスなので学習用データは 100 クラスであり、特徴量空間中にクラスの代表点が 100 個存在することになる。この代表点同士がなす角度を、学習率を変化させるタイミングで全代表点同士について計算する。図 5.2 に代表点同士の cosine 類似度の分布変化を示す。スケーリングパラメタを 6.5 と 20、特徴量空間の次元数を 64 と 512 にそれぞれ設定して学習を進めた際の変化である。また表 5.1 に、図 5.2 と同順の条件で各 epoch における cosine 類似度の分散の大きさを示す。スケーリングパラメタが 20 の時のみ、提案手法である線形減少スケーリングパラメタによる追加学習も行った。この追加学習の際の cosine 類似度分布は付録の図 A.3 に掲載する。

スケーリングパラメタが十分大きい定数であるとき、学習が進んでいる間この代表点間の角度は変化しておらず、分散が約 0.015 程度で cosine は-0.4 から 0.4 程度である初期化時から大きな変化はなかった。しかしスケーリングパラメタが小さい定数の時、1epoch の学習を終えて分散が 0.040 と大きくなり、cosine の値の分布も両端が 0.6(約 55 度) から-0.6(約 125 度) 程度になっていた。学習が進むに従って分散は小さくなっていくが、cosine 類似度の大きさが最大 0.6 程度の距離である 2 代表点が残っていた。このスケーリングパラメタの値で学習を進めた時には最終的に評価スコアは上がっていない (図 4.5 と表 4.4 および 4.5)。また、特徴量の次元が高い時には小さなスケーリングパラメタでも代表点同士の cosine 類

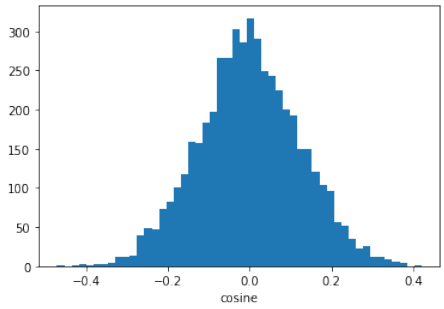
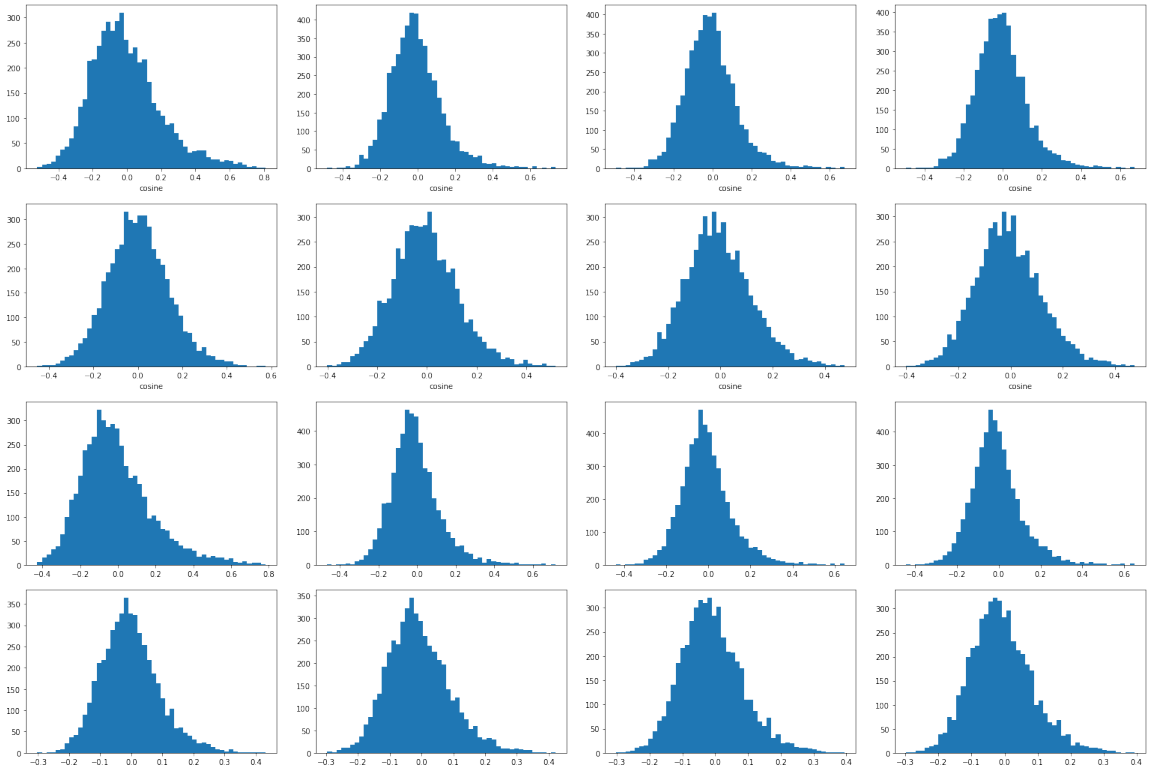


図 5.1　クラスの代表点ベクトルを初期化した際の cosine 類似度の分布。



(a)　(b)　(c)　(d)

図 5.2　CUB-200-2011 データセットの学習用クラスについて、スケーリングパラメタ s と特徴量次元数 d を変えて学習した時の代表点同士の cosine 類似度の分布変化。(a)1epoch、(b)20epoch、(c)40epoch、(d)50epoch

似度はほぼ 0 に収まっていた。

この結果から、学習初期に十分大きなスケーリングパラメタで損失関数の値を大きくしなければ、特徴量空間中でクラス同士が少なくとも一時的には近づいてしまうことがわか

表 5.1　CUB-200-2011 データセットの学習用クラスについて、スケーリングパラメタ s と特徴量次元数 d を変えて学習した時の代表点同士の cosine 類似度の分散の変化。

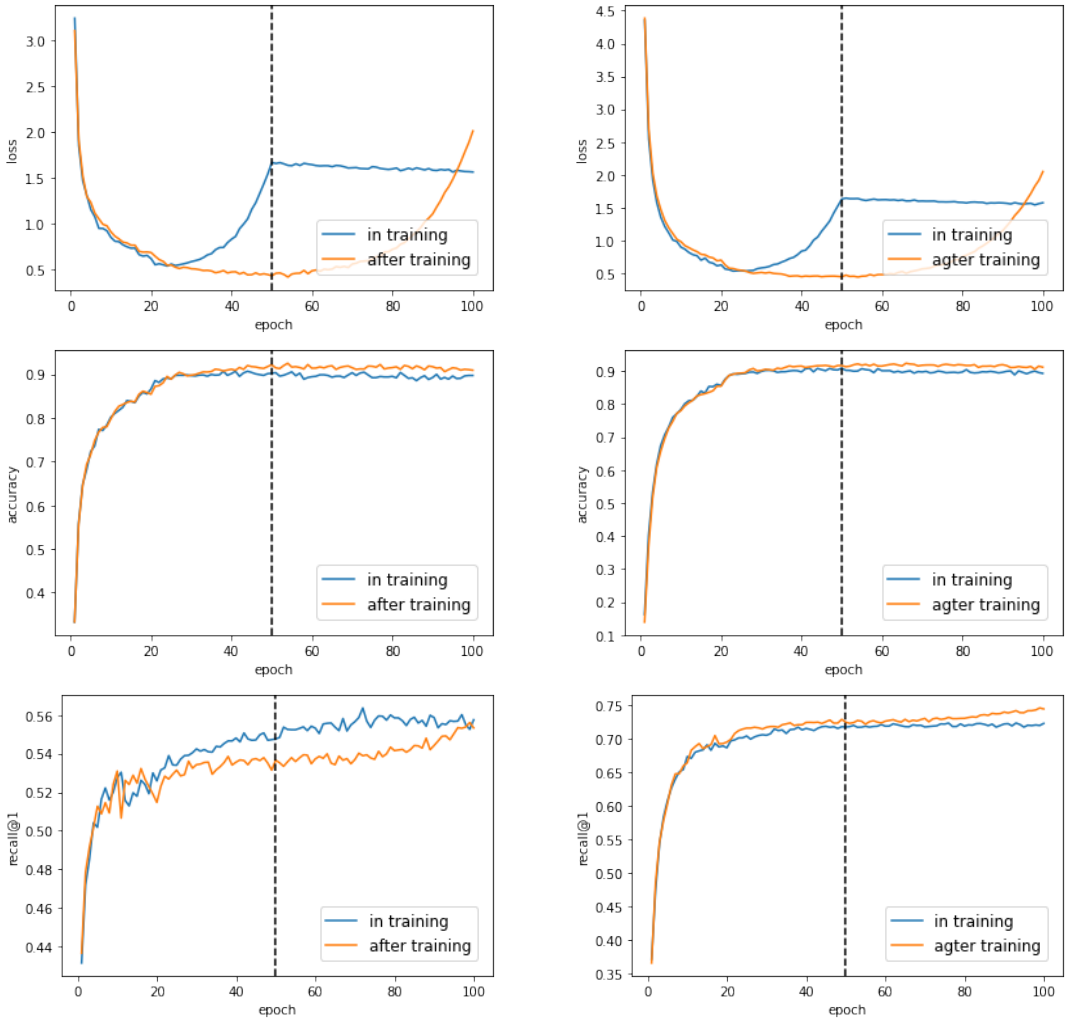
変数	epoch					
	1	20	40	50	75	100
$s = 6.5, d = 64$	0.0409	0.0189	0.0172	0.0171	-	-
$s = 6.5, d = 512$	0.0189	0.0171	0.0161	0.0160	0.0158	0.0168
$s = 20, d = 64$	0.0353	0.0163	0.0147	0.0145	-	-
$s = 20, d = 512$	0.0088	0.0096	0.0089	0.0088	0.0089	0.0108

る。十分に大きなスケーリングパラメタには他クラスのデータを遠ざけるように学習を進める役割があるため、小さな値では代表点同士が近づいてしまうためであると考えられる。

5.2　通常学習時にスケーリングパラメタを線形に減少させる手法

上記の実験より、評価スコアが上がり続けるような学習を進めることができるかどうかは学習初期に決まってしまうのではないかと考え、線形減少するスケーリングパラメタを学習初期から導入する実験を行った。通常の学習開始時から 1epoch 毎に最適なスケーリングパラメタを算出して減少させていく手法も提案されている [6] が、クラスの代表点間の角度やミニバッチ内のデータ特徴量ベクトル間の cosine 類似度を逐次算出しなければならず、計算コストが非常に高くなってしまう。学習開始時からスケーリングパラメタを線形に減少させるという簡易な実装によりこの手法を近似することで、計算コストを増大させることなくより識別能力を獲得できる学習が進められるかどうかを確認する。スケーリングパラメタの初期値を 20 に設定し、最初から 1epoch ごとに線形減少させながら 50epoch 目に 5 になるように学習を行った。この結果を、通常学習後に線形減少するスケーリングパラメタで追加学習した結果と比較して図 5.3 と表 5.2 に示す。上段が CUB-200-2011、下段が Cars-196 データセットの結果である。

図 5.3(a) から分かるように、スケーリングパラメタが小さくなると損失関数の値は大きくなる。図 3.5 に表されているように、スケーリングパラメタが小さくなるとクラスの代表点となす角度の決定境界がより小さくなり、同じ角度で位置する特徴量ベクトルについて損失関数の値が大きく計算されるようになるためである。学習開始時からスケーリングパラメタを小さくしていく手法では損失関数の値は急激に大きくなり、追加の学習が進むに従って少しずつ小さくなっている。一方で線形に減少するスケーリングパラメタでは、追



(a)　(b)

図 5.3　通常学習と追加学習それぞれでスケーリングパラメタを減少させた実験における損失関数の値、分類精度、Recall@1 スコアの変化。(a)CUB-200-2011、(b)Cars-196

加学習の後半まで損失関数の値を小さく抑えられている。評価スコアについては追加学習時に減少させる手法が高くなっており、学習初期からスケーリングパラメタを変化させる操作は不安定な学習をもたらすと考えられる。別のデータセットで学習を行う際にはテストデータが用意できないという状況も考えられるため、1epoch 毎に評価スコアを計測することは困難であると考えて良い。その場合に学習の初期段階から重要なパラメタを変化させるリスクは大きく、既に学習が完了しているネットワークモデルに対しても追加学習を行うことができるという点でも本研究の提案手法には有効性がある。

表 5.2　追加実験 2 における通常学習と追加学習終了時の分類精度と Recall@1 スコア。

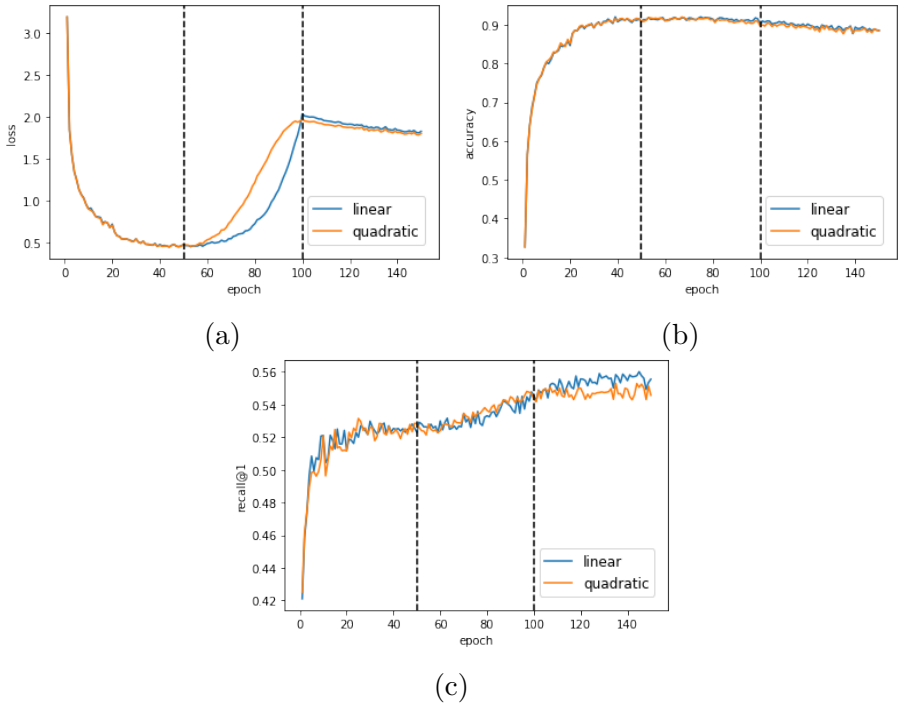
	減少させるタイミング	分類精度		Recall@1	
		50epoch	100epoch	50epoch	100epoch
CUB-200-2011	通常学習時	0.9019	0.8970	0.5476	0.5575
	追加学習時	0.9216	0.9090	0.5366	0.5533
Cars-196	通常学習時	0.9085	0.8927	0.7180	0.7234
	追加学習時	0.9165	0.9113	0.7252	0.7450

スケーリングパラメタが最終値に近づくにつれて損失関数の値が指数関数的に大きくなっているという結果から、スケーリングパラメタが 10 未満のような小さな値のときに学習の難易度が大きく変化しているのではないかと考え、次節の実験を行った。具体的には追加学習の際に、最初に急激にスケーリングパラメタを減少させて徐々に減少の度合いが小さくなるように、線型ではなく下に凸な二次関数的な変化のさせ方を用いた。

5.3　2 次関数的に減少するスケーリングパラメタ

図 5.4 に、二次関数的にスケーリングパラメタを減少させて追加の学習を行った結果を示す。なお、スケーリングパラメタの減少を止めた後にさらに 50epoch 追加の学習を行った。またこの実験は CUB-200-2011 データセットのみで行った。

損失関数の値については、追加学習における極大値を少しではあるが小さく抑えられている。具体的には、線形減少の極大値が 100epoch 目の 2.033 に対して二次関数的減少の極大値が 99epoch 目の 1.972 であった。accuracy について差は無かった。Recall@1 スコアはスケーリングパラメタが減少しているときはどちらも同じような値であったが、さらに小さくなったスケーリングパラメタのまま追加学習を行うと線形減少させた方がスコアがさらに高くなった。この結果については、損失関数の値が再び大きくなっていく際の学習が初期段階の学習のように Recall@1 スコアを高める方向に進むのではないかと考えられるが、より詳細な実験や理論的なアプローチを考えていく必要があり、現在その方針については検討中である。また一連の実験結果よりスケーリングパラメタは最終値が重要であり、追加の学習を十分に進めることで減らし方には依存しないという結果が得られているが、この考察についてはさらに詳細な検討を行う必要がある。



(a)　(b)

(c)

図 5.4　下に凸な二次関数に従って減少するスケーリングパラメタを用いた学習結果。(a) 損失関数、(b)accuracy、(c)Recall@1

第6章　結論

6.1　結論

近年の Deep Metric Learning において有効な手法である cosine 類似度を用いたクラス分類による学習において、Softmax 関数への入力となる cosine 類似度を定数倍するスケーリングパラメタに関する考察と実験を行った。学習中にテストスコアを計測することで、学習が進むかどうかはスケーリングパラメタの値に大きく依存していることが分かった。最適なスケーリングパラメタを算出する既存研究について、クラス数が少ないデータセットに対して当てはまらないことを確認した。

またスケーリングパラメタを線形に減少させながら追加の学習を行う提案手法では、定数を用いて学習を終える場合に比べて評価スコアである Recall@1 が向上することを確認した。この追加学習の際には最終値によって評価スコアが異なり、良いスコアを獲得できる値の範囲があることを示した。またこの線形減少させる手法に加えて追加学習の開始時にスケーリングパラメタを小さな値に切り替える手法を提案および比較し、どちらもスコアが向上することを確認した。

さらに議論を進めるために 3 つの実験を行った。学習中にデータセットのクラスを表す代表点ベクトルが特徴量空間中でなす角度の分布について調べ、スケーリングパラメタが小さい時には初期化時から互いに近づいてしまう代表点があることが分かった。つまり cosine 類似度をはじめとした値域の幅が小さい距離指標を用いた Metric Learning では、スケーリングパラメタを用いることで学習をうまく進めることができる。追加学習ではなく学習開始時からスケーリングパラメタを減少させる手法では、追加の学習時に減少させていく方が評価スコアが高いことを確認した。追加学習時にスケーリングパラメタを下に凸な二次関数的に減少させる手法では、その後十分に学習を進めることで評価スコアに差がないことを確認した。

6.2　今後の展望

スケーリングパラメタに直接関連すると考えられるマージンや学習率を変化させる実験により、最適な学習手法に関する理論を構築していきたい。またスケーリングパラメタを変化させることの学習への影響について、数式的な理論に基づく解釈を可能にしたい。

さらに Metric Learning は Adversarial Attack、異常検知、推薦といった機械学習分野の様々なタスク、また自然言語処理や音声など画像認識以外の分野においても応用研究が多数提案されている。これらについても改善手法や新規手法を詳細に検討していきたい。

謝辞

指導教員である工藤知宏教授には、最初から最後まで有益な助言を頂きご指導を賜りました。産業技術総合研究所の池上努博士には、豊富な研究活動のご経験から様々な助言をしていただきました。長期でインターンをさせていただいた株式会社 ABEJA の藤本敬介様には、研究テーマに関連した論文の紹介やその読み方など、本研究の土台を築く御助力をいただきました。工藤研究室の皆様には、ご指導ご鞭撻を賜りました。また家族には、常に温かい応援をしていただき励みになりました。支えてくださった皆様に感謝いたします。

発表文献

国内会議 (査読なし)

1. 佐藤 啓樹, 池上 努, 藤本 敬介, 工藤 知宏. “Softmax 関数への動的スケーリングパラメタの導入による Deep Metric Learning の精度向上”, 電子情報通信学会ニューロコンピューティング研究会 (Oct.2020).

参考文献

[1] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, Vol. 86, No. 11, pp. 2278–2324, 1998.

[2] A. Bellet, A. Habrard, and M. Sebban. *Metric Learning*. Morgan & Claypool, 2015.

[3] Xiao Zhang, Rui Zhao, Junjie Yan, Mengya Gao, Yu Qiao, Xiaogang Wang, and Hongsheng Li. P2sgrad: Refined gradients for optimizing deep face models. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 9906–9914, 2019.

[4] Byungsoo Ko and Geonmo Gu. Embedding expansion: Augmentation in embedding space for deep metric learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7255–7264, 2020.

[5] Eu Wern Teh, Terrance DeVries, and Graham W Taylor. Proxynca++: Revisiting and revitalizing proxy neighborhood component analysis. *arXiv preprint arXiv:2004.01113*, 2020.

[6] Xiao Zhang, Rui Zhao, Yu Qiao, Xiaogang Wang, and Hongsheng Li. Adacos: Adaptively scaling cosine logits for effectively learning deep face representations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 10823–10832, 2019.

[7] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The Caltech-UCSD Birds-200-2011 Dataset. Technical report, 2011.

[8] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *4th International IEEE Workshop on 3D Representation and Recognition (3dRR-13)*, Sydney, Australia, 2013.

[9] Qi Qian, Lei Shang, Baigui Sun, Juhua Hu, Hao Li, and Rong Jin. Softtriple loss: Deep metric learning without triplet sampling. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 6450–6458, 2019.

[10] Sungyeon Kim, Dongwon Kim, Minsu Cho, and Suha Kwak. Proxy anchor loss for deep metric learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3238–3247, 2020.

[11] Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou. Arcface: Additive angular margin loss for deep face recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4690–4699, 2019.

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

[13] Mahmut Kaya and Hasan Şakir Bilge. Deep metric learning: A survey. *Symmetry*, Vol. 11, No. 9, p. 1066, 2019.

[14] Yandong Wen, Kaipeng Zhang, Zhifeng Li, and Yu Qiao. A discriminative feature learning approach for deep face recognition. In *European conference on computer vision*, pp. 499–515. Springer, 2016.

[15] Kihyuk Sohn. Improved deep metric learning with multi-class n-pair loss objective. In *Advances in neural information processing systems*, pp. 1857–1865, 2016.

[16] Elad Hoffer and Nir Ailon. Deep metric learning using triplet network. In *International Workshop on Similarity-Based Pattern Recognition*, pp. 84–92. Springer, 2015.

[17] Raia Hadsell, Sumit Chopra, and Yann LeCun. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, Vol. 2, pp. 1735–1742. IEEE, 2006.

[18] Ben Harwood, Vijay Kumar BG, Gustavo Carneiro, Ian Reid, and Tom Drummond. Smart mining for deep metric learning. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2821–2829, 2017.

[19] Qi Qian, Rong Jin, Jinfeng Yi, Lijun Zhang, and Shenghuo Zhu. Efficient distance metric learning by adaptive sampling and mini-batch stochastic gradient descent (sgd). *Machine Learning*, Vol. 99, No. 3, pp. 353–372, 2015.

[20] Yair Movshovitz-Attias, Alexander Toshev, Thomas K Leung, Sergey Ioffe, and Saurabh Singh. No fuss distance metric learning using proxies. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 360–368, 2017.

[21] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 815–823, 2015.

[22] Feng Wang, Xiang Xiang, Jian Cheng, and Alan Loddon Yuille. Normface: L2 hypersphere embedding for face verification. In *Proceedings of the 25th ACM international conference on Multimedia*, pp. 1041–1049, 2017.

[23] Rajeev Ranjan, Carlos D Castillo, and Rama Chellappa. L2-constrained softmax loss for discriminative face verification. *arXiv preprint arXiv:1703.09507*, 2017.

[24] Jian Wang, Feng Zhou, Shilei Wen, Xiao Liu, and Yuanqing Lin. Deep metric learning with angular loss. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2593–2601, 2017.

[25] Weiyang Liu, Yandong Wen, Zhiding Yu, Ming Li, Bhiksha Raj, and Le Song. Sphereface: Deep hypersphere embedding for face recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 212–220, 2017.

[26] Hao Wang, Yitong Wang, Zheng Zhou, Xing Ji, Dihong Gong, Jingchao Zhou, Zhifeng Li, and Wei Liu. Cosface: Large margin cosine loss for deep face recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5265–5274, 2018.

[27] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

[28] Xu Zhang, Felix X. Yu, Svebor Karaman, Wei Zhang, and Shih-Fu Chang. Heated-up softmax embedding. *CoRR*, Vol. abs/1809.04157, , 2018.

[29] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *nature*, Vol. 323, No. 6088, pp. 533–536, 1986.

[30] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[31] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1–9, 2015.

[32] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*, pp. 448–456, 2015.

[33] Prasanta Chandra Mahalanobis. On the generalized distance in statistics. National Institute of Science of India, 1936.

[34] Kilian Q Weinberger and Lawrence K Saul. Distance metric learning for large margin nearest neighbor classification. *Journal of Machine Learning Research*, Vol. 10, No. 2, 2009.

[35] Chengzhi Mao, Ziyuan Zhong, Junfeng Yang, Carl Vondrick, and Baishakhi Ray. Metric learning for adversarial robustness. In *Advances in Neural Information Processing Systems*, pp. 480–491, 2019.

[36] Luke Ditria, Benjamin J. Meyer, and Tom Drummond. Opengan: Open set generative adversarial networks, 2020.

[37] Maximillian Nickel and Douwe Kiela. Poincaré embeddings for learning hierarchical representations. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, Vol. 30, pp. 6338–6347. Curran Associates, Inc., 2017.

[38] Shuo Chen, Lei Luo, Jian Yang, Chen Gong, Jun Li, and Heng Huang. Curvilinear distance metric learning. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-

Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, Vol. 32, pp. 4223–4232. Curran Associates, Inc., 2019.

[39] Hyun Oh Song, Yu Xiang, Stefanie Jegelka, and Silvio Savarese. Deep metric learning via lifted structured feature embedding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4004–4012, 2016.

[40] Wenzhao Zheng, Zhaodong Chen, Jiwen Lu, and Jie Zhou. Hardness-aware deep metric learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 72–81, 2019.

[41] Yuge Huang, Yuhan Wang, Ying Tai, Xiaoming Liu, Pengcheng Shen, Shaoxin Li, Jilin Li, and Feiyue Huang. Curricularface: adaptive curriculum learning loss for deep face recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5901–5910, 2020.

[42] Nicolas Aziere and Sinisa Todorovic. Ensemble deep manifold similarity learning using hard proxies. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7299–7307, 2019.

[43] Daniel Miller, Ira Kemelmacher-Shlizerman, and Steven M. Seitz. Megaface: A million faces for recognition at scale. *CoRR*, Vol. abs/1505.02108, , 2015.

[44] Yu Liu, Hongyang Li, and Xiaogang Wang. Learning deep features via congenerous cosine loss for person recognition. *arXiv preprint arXiv:1702.06890*, 2017.

[45] Weiyang Liu, Yandong Wen, Zhiding Yu, and Meng Yang. Large-margin softmax loss for convolutional neural networks. In *ICML*, Vol. 2, p. 7, 2016.

[46] Wei-Feng Ou, Lai-Man Po, Chang Zhou, Yu-Jia Zhang, Li-Tong Feng, Yasar Abbas Ur Rehman, and Yu-Zhi Zhao. Lincos-softmax: Learning angle-discriminative face representations with linearity-enhanced cosine logits. *IEEE Access*, Vol. 8, pp. 109758–109769, 2020.

[47] James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *The Journal of Machine Learning Research*, Vol. 13, No. 1, pp. 281–305, 2012.

[48] Jasper Snoek, Hugo Larochelle, and Ryan P Adams. Practical bayesian optimization of machine learning algorithms. *Advances in neural information processing systems*, Vol. 25, pp. 2951–2959, 2012.

[49] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in neural information processing systems*, pp. 8026–8037, 2019.

[50] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

[51] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[52] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pp. 249–256, 2010.

付 録A　付録

図 A.1　GoogleNet アーキテクチャの全体図。

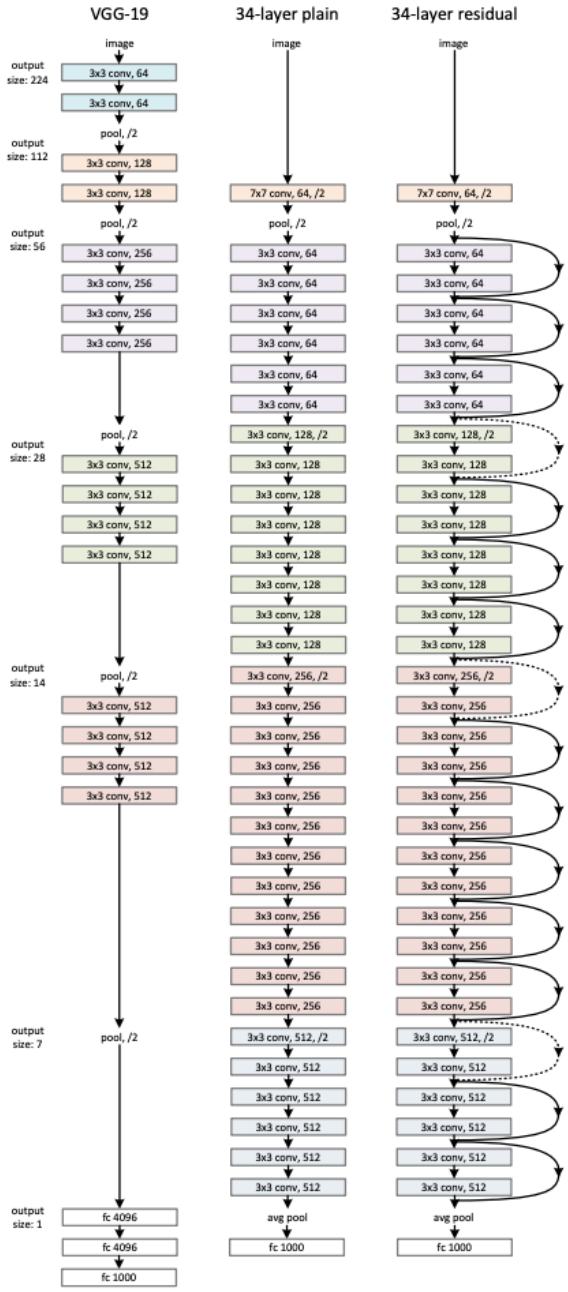
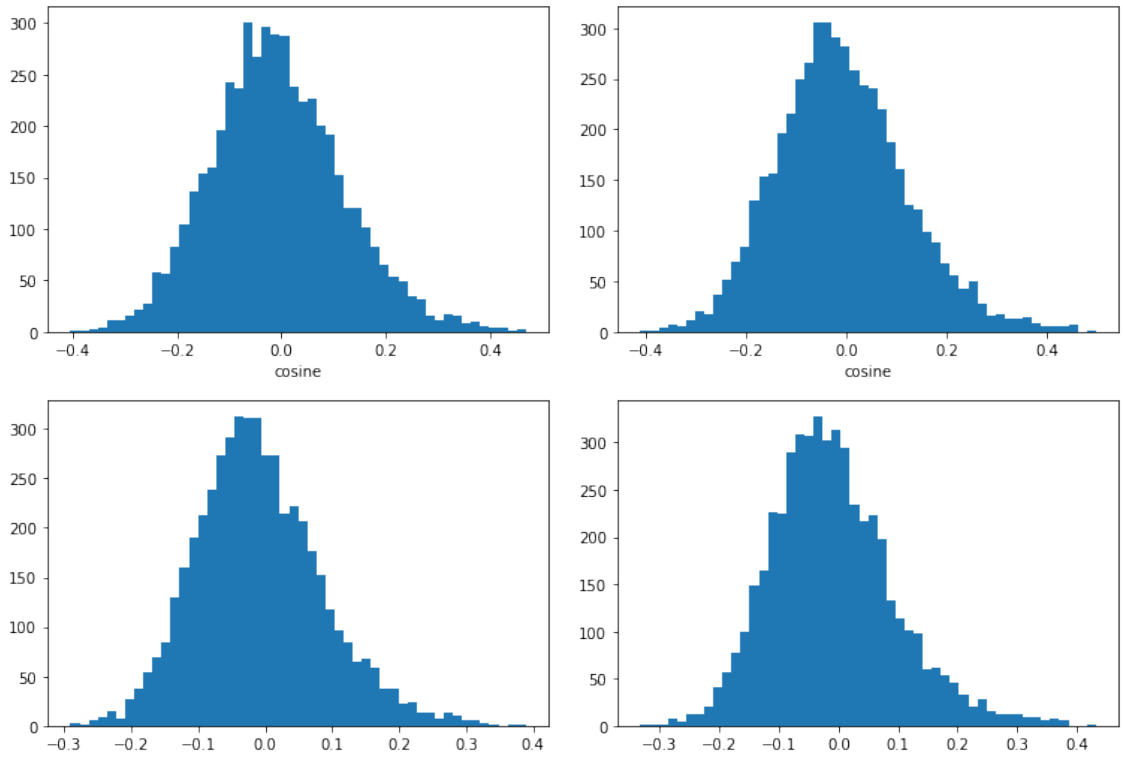


図 A.2　VGG アーキテクチャと比較した ResNet アーキテクチャの全体図。



(a)　(b)

図 A.3　線形減少するスケーリングパラメタで追加学習を行った際の、特徴量次元数 64(上段) と 512(下段) それぞれにおける cosine 類似度の分布変化。(a)75epoch、(b)100epoch