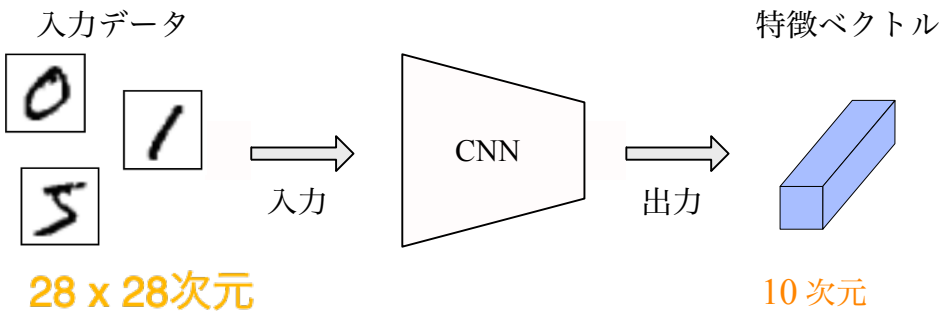
距離学習 (Metric learning)

ディープラーニング技術を用いた距離学習は、人物同定 (Person Re-Identification) をはじめ、顔認識、画像分類、画像検索、異常検知など幅広いタスクに利用される技術となっています。この解説プリントでは、距離学習の基本的なアイディア、そして深層距離学習の代表的な手法である Siamese Network [Hadsell+, "Dimensionality Reduction by Learning an Invariant Mapping", CVPR (2006)]、および、Triplet Network [Hoffer+,"Deep metric learning using Triplet network", arXiv:1412.6622 (2014)] について説明していきます。

♣ 距離学習とは

距離学習ではデータ間の **metric**、すなわち「データ間の距離」を学習します。データ間の距離を適切に測ることができれば、距離が近いデータ同士をまとめてクラスタリング[1]ができたり、他のデータ要素から距離が遠いデータを異常と判定することで異常検知したりと様々な応用が可能となります。距離学習自体は古くからある手法ですが、近年のディープラーニングの発展とともに、ディープラーニング技術を利用した距離学習の手法が数多く提案されています。このような手法は、特に**深層距離学習 (deep metric learning)** と呼ばれています。

一般に、画像や音声などの多次元データはニューラルネットワークを用いることにより、次元削除 (データ圧縮) することが出来ます。例えば、畳み込みニューラルネットワーク (Convolutional Neural Network: CNN) を用いた画像分類の場合、28×28 サイズの画像をニューラルネットワーク (CNN) に入力することで 10 次元のベクトルに圧縮することができます。つまり、元々 $28 \times 28 = 784$ 個の数値で表現されていた画像の情報を 10 個の変数に圧縮したと解釈できます。CNN から出力されるベクトルは、入力データの重要な情報を抽出したベクトルと考えられるため、一般に**特徴ベクトル (feature vector)** と呼ばれます。

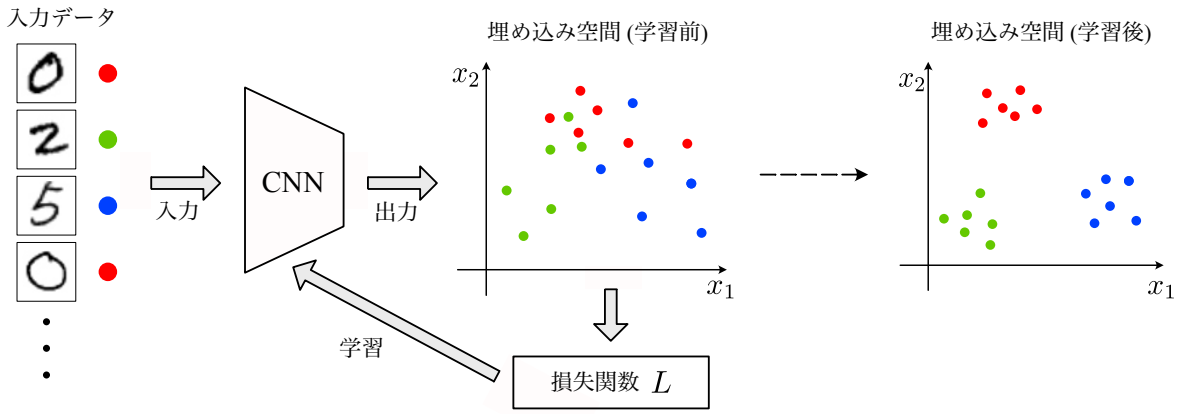


ただし、未学習のネットワークにデータを入力しても出力されるのは無意味なベクトルであり、特徴ベクトルに意味を持たせるには何らかの手法で CNN を学習する必要があります。深層距離学習はその 1 つの方法であり、2 つの特徴ベクトル間の距離がデータの類似度を反映するようにネットワークを学習します。具体的には、

[1]機械学習の一種で、データ間の類似度に基づいてデータをグループ分けする手法のことです。

- 同じクラスに属する (=類似) サンプルから得られる特徴ベクトルの距離間は小さくする
- 異なるクラスに属する (=非類似) サンプルから得られる特徴ベクトルの距離間は大きくする

といった具合です。特徴ベクトルの属する空間は**埋め込み空間 (embedding space)**[2]と呼ばれますが、この埋め込み空間内で類似サンプルの特徴ベクトルは近くに、非類似サンプルの特徴ベクトルは遠くに配置されるように学習を行うことになります。つまり、CNN を深層距離学習の手法を用いて学習していくと、類似サンプル (から得られる特徴ベクトル) は埋め込み空間内で密集していき、逆に非類似サンプルは離れていきます (下図参照)。



この図では、特徴ベクトルが 2 次元のベクトル (x_1, x_2) であるとして、特徴ベクトルを埋め込み空間上の点として可視化しています。学習後のような埋め込み空間を構成することができれば、画像分類やクラスタリングが容易になるのは一目瞭然です。もちろん、画像分類などのタスクに対する精度の向上は、ニューラルネットワークモデル自体の構造を工夫 (複雑化) することでも達成できるでしょう。しかし、ネットワークの構造自体は工夫しなくても、類似度を反映した埋め込み空間を構成できるように学習を行うだけで精度向上が可能になる、というのが深層距離学習のアプローチとなります。

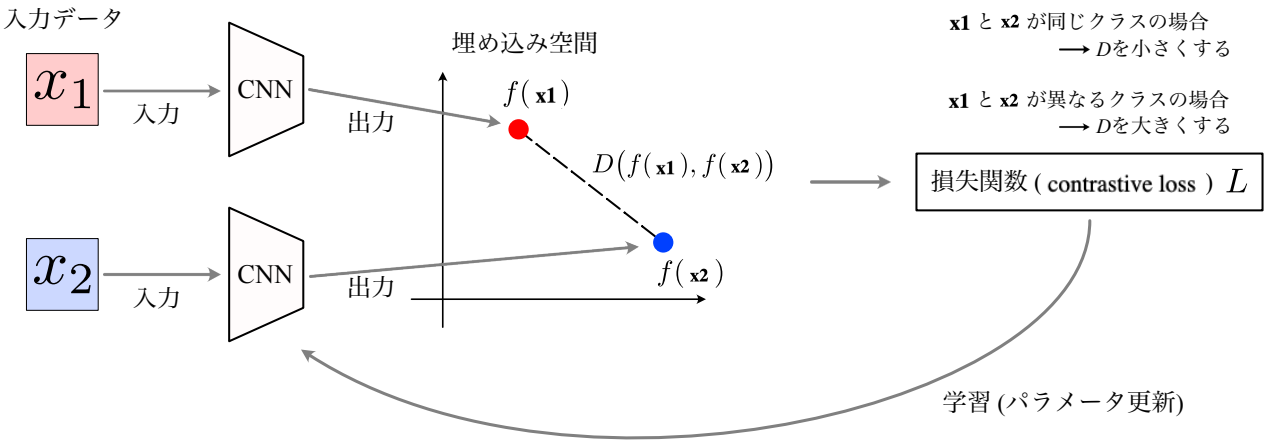
では、深層距離学習ではどのようにしてサンプル間の類似度を反映した埋め込み空間を構成するのでしょうか？ 以下では、深層距離学習の代表的な手法である Siamese network と Triplet network について説明してきます。

♣ 手法 1：Siamese network (シャムネットワーク)

Siamese network は、2006 年に提案された深層距離学習の中でも比較的歴史の古い手法です。Siamese network の特徴は、2 つのサンプルをペアで入力しそれらのサンプル間の距離を明示的に表現して調整する点で、次のような構造になっています[3]。

[2]特徴量空間や特徴空間などと呼ばれることもあります。

[3]以下では入力データとして画像を想定し、ネットワーク部分を CNN としていますが、Siamese network や Triplet network の方法はネットワークの詳細には依らず、通常の深層ニューラルネットワーク (Deep Neural Network: DNN) などに対しても適用できます。



上図に示すように、CNN[4]にペア画像 (それぞれの画像のデータを x_1、x_2 とする) を入力し、その出力である特徴ベクトルを $f(\mathbf{x1})$、$f(\mathbf{x2})$ とします。そして埋め込み空間内での 2 つの特徴ベクトルの距離 $D\big(f(\mathbf{x1}), f(\mathbf{x2})\big)$ が「最適」となるようにネットワークのパラメータを学習します。つまり、ペア画像が同じクラスの場合には距離 D が小さくなるように、逆に異なるクラスの場合には大きくなるように損失関数 L を設計し、学習を行います。

では、具体的にどのような損失関数を用いれば、最適な距離 D を実現する CNN を学習できるのでしょうか？ Siamese network では、次の損失関数を最小化することで学習を行います。

$$L = \frac{1}{2}\big[yD^2 + (1-y)\max(m-D, 0)^2\big]$$

この損失関数は、特に **contrastive loss** と呼ばれます。上式で、D は埋め込み空間での $f(\mathbf{x1})$ と $f(\mathbf{x2})$ の距離を表します[5]。このような 2 点間の距離の測り方には様々な方法がありますが、提案論文では次のユークリッド距離 (L2 距離) が用いられています[6]。

$$D = \|f(\mathbf{x1}) - f(\mathbf{x2})\|_2$$

contrastive loss の変数には、この距離 D の他に、y と m の 2 つのパラメータが含まれています。これらのパラメータについては後ほど説明しますが、これらは学習時には定数となります。つまり、contrastive loss の実質的な変数は距離 D のみであり、contrastive loss を最適化するためには距離 D を適切に調整 (学習) する必要があります。そして、距離 D は CNN の出力値 (特徴ベクトル) から計算されるため、距離 D を学習することは間接的に CNN のパラメータを学習していることに相当します。

唐突に導入した contrastive loss ですが、一体どのような意味を持っているのでしょうか？ contrastive loss は、次のように 2 つの項 $[yL_{\mathrm{I}}$ と $(1-y)L_{\mathrm{II}}]$ から構成されています。

---

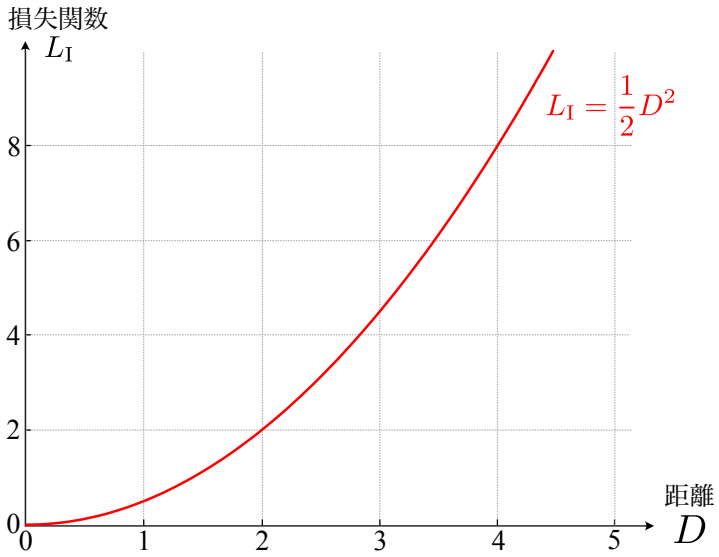
[4] 2 つの CNN は共通で、パラメータを共有しています。
[5] 丁寧に書くと $D\big(f(\mathbf{x1}), f(\mathbf{x2})\big)$ ですが、式が見づらくなるので引数を落として単に D としています。
[6] 埋め込み空間の 2 つのベクトルの距離を計測できる関数であれば、ユークリッド距離以外でも大丈夫です。

$$L = yL_1 + (1-y)L_2$$

$$L_{\mathrm{I}} = \frac{1}{2}D^2$$

$$L_{\mathrm{II}} = \frac{1}{2}\max(m-D,0)^2$$

以下では、L_{I} と L_{II} の意味、そして係数 y と $(1-y)$ の役割について説明していきます。先ず L_{I} ですが、この損失関数は埋め込み空間で 2 つの入力データの位置を近づける効果を持っています。L_{I} は 2 次関数で、距離 D が大きくなるほど値が大きくなる関数となっています。

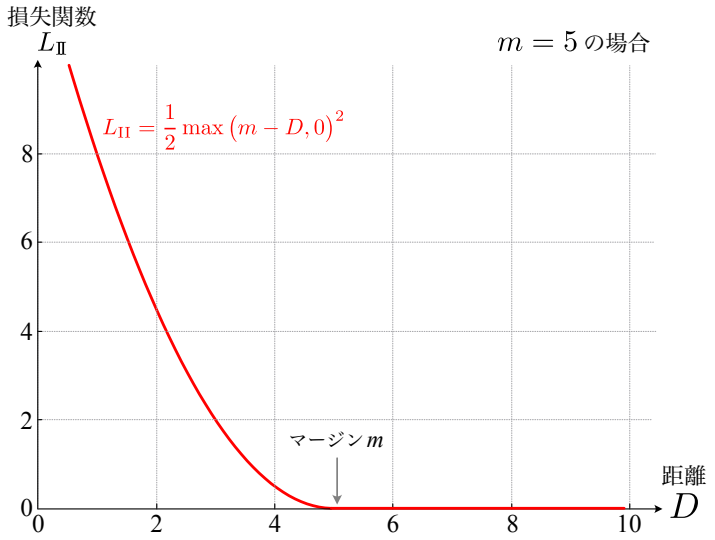


この損失関数では、2 つのデータ間の距離 D が大きくなるほど値が大きくなっており (=ペナルティーが課されており)、この損失関数を最適化することは、2 つのデータ間の距離 D を小さくする、つまり埋め込み空間で 2 つのデータを近づけることに対応します。尚、2 次関数でなくても距離 D が大きくなるほど値が大きくなるような関数は無数にありますが、Siamese network では 2 つのデータを近づけるために上のような 2 次関数を用いることになっています。

次に L_{II} ですが、これは埋め込み空間で 2 つの入力データの特徴ベクトルを離す効果を持ちます。L_{II} に含まれる max 関数は、2 つの引数 $(m-D$ と $0)$ を比較して大き方を出力する関数です。つまり、

$$L_{\mathrm{II}} = \frac{1}{2}\max(m-D,0)^2 = \begin{cases} \frac{1}{2}(m-D)^2 & (m-D>0 \Leftrightarrow D<m \text{ の場合}) \\ 0 & (m-D\leq 0 \Leftrightarrow D\geq m \text{ の場合}) \end{cases}$$

となります。m は**マージン (margin)** と呼ばれるパラメータで、このマージンと 2 つのデータの埋め込み空間内での距離 D の大小関係に応じて、場合分けが生じています。この場合分けに注意して L_{II} のグラフを描くと、次のようになります (下図では $m=5$ としています)。



上図から分かるように、この損失関数は距離 D が小さくなるほど値が大きくなりペナルティーが課される構造になっています。そのため、この損失関数を最適化することは距離 D を大きくする、つまり 2 つの入力データを遠ざけることに対応します。ただし、D が大きくなるにつれて損失関数が無限に小さくなり続けてしまうと、埋め込み空間でのデータ間の位置が無限に離れてしまいます。これを防ぐために、マージン m 以上距離が離れたら (上図の場合は $D \geq 5$) ペナルティーを課すのをやめる (=損失関数の値を 0 とする) ようになっています。マージン m は、どの程度の距離までペナルティーを課すかを決めるパラメータになっています。

さて、2 つの損失関数 L_{I} と L_{II} の役割について整理しましょう。

- 損失関数 L_{I} ：埋め込み空間での入力データ間の距離 D を小さくする
- 損失関数 L_{II}：埋め込み空間での入力データ間の距離 D を大きくする

前のセクションで述べたように、深層距離学習では埋め込み空間で類似データは近くに、非類似データは遠くに配置されるようにネットワークを学習します。そのために、Siamese network ではペアで入力したデータが同じクラスである場合には L_{I} を、異なるクラスである場合には L_{II} を損失関数として用いることで距離学習を行います。

$$L = \begin{cases} L_{\mathrm{I}} = \dfrac{1}{2}D^2 & \text{(入力ペアが同じクラスの場合)} \\ L_{\mathrm{II}} = \dfrac{1}{2}\max(m - D, 0)^2 & \text{(入力ペアが異なるクラスの場合)} \end{cases}$$

これを 1 つの式で表現するために、**識別ラベル**と呼ばれる変数 y を導入します。この識別ラベルは、入力サンプルのペアが同じクラスに属する場合には $y = 1$、異なるクラスに属する場合には $y = 0$ という値を取ります。この識別ラベルを用いると上の損失関数は次のように 1 つの式にまとめることができます。

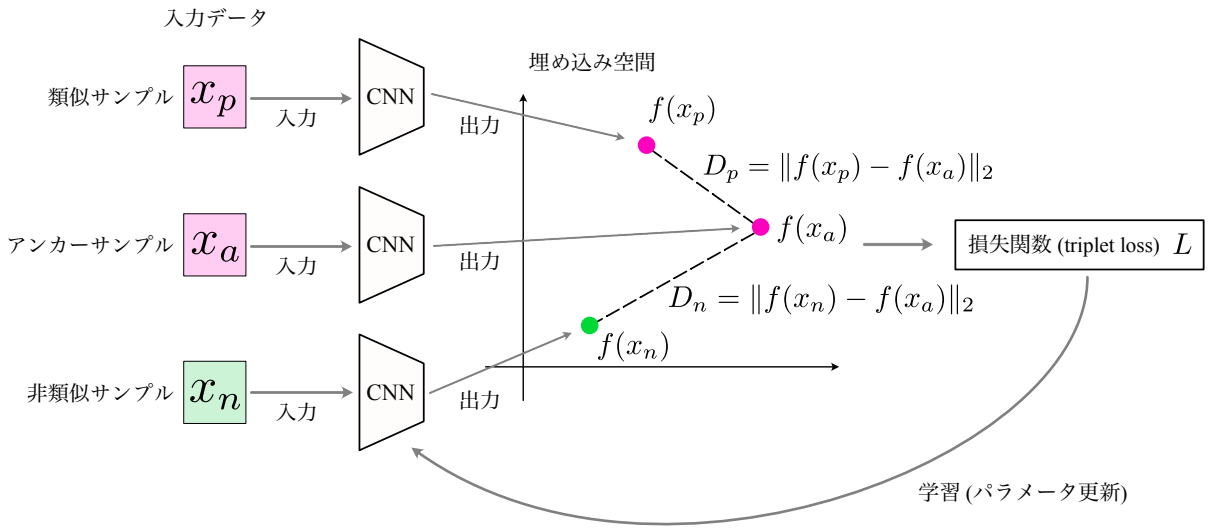
$$L \;=\; yL_1 + (1 - y)L_2 \;=\; \frac{1}{2}\left[yD^2 + (1 - y)\max(m - D, 0)^2\right]$$

入力サンプルが同じクラスの場合には $y=1$ となるため、2項目の $(1-y)L_2$ は0となって $L=L_1$ となります。逆に異なるクラスの場合には $y=0$ となるため、1項目の yL_1 は0となって $L=L_2$ となっています。以上のように、contrastive loss は、埋め込み空間でのデータ間の距離を近づける損失関数 L_{I}、そしてマージン m 以上の距離まで遠ざける損失関数 L_{II} から構成されており、識別ラベル y を用いることで入力データに応じて損失関数 L_{I} と L_{II} が切り替わるような構造になっています。

ここまでの Simese netwok で用いる contrastive lossについて説明してきましたが、この損失関数には大きな欠点があります。それは、異なるクラスのデータに対しては距離 D がマージン m を超えた時点で最適化が終了するのに対し、同じクラスのデータに対しては $D=0$ となるまで、つまり埋め込み空間中のある1点に埋め込まれるまで最適化し続けてしまう点です。この不均衡により、類似度を反映した良い埋め込み空間を得ることが難しい傾向にあります。そこで考案されたのが、次のセクションで説明する Triplet network となります。

♣ 手法2：Triplet network (トリプレットネットワーク)

Triplet network は、2014年に提案された深層距離学習の手法です[7]。Siamese network では2つのサンプルをセットにして入力するのに対して、Triplet network では3つのサンプルを一組で入力するようになっています[8]。Triplet network の構造を表すと、次の図のようになります。



Triplet network では Siamese network 同様、埋め込み空間で同じクラスのデータは近くに、異なるクラスのデータは遠くに配置されるように損失関数を設計していますが、入力データの準備方法が異なります。入力データのセットは次の手順で準備します。

1. 基準となるサンプル x_a (アンカーサンプル) を選択する。
2. アンカーサンプル x_a の類似サンプル x_p と非類似サンプル x_n を1つずつ選択する[9]。

[7]原論文では、画像検索における順位付けを学習する目的で考案されています。
[8]そもそも “Triplet” というのが「3つ組」のような意味を持っています。
[9]x_p と x_n の p と n は、それぞれ positive と negative の頭文字です。

3つの入力データのセットを準備したら、後は各サンプルを同じ CNN に通して埋め込み空間へマップし、損失関数 L を計算することになります。入力 x_a, x_p, x_n に対する CNN の出力 (特徴ベクトル) をそれぞれ $f(x_a)$, $f(x_p)$, $f(x_n)$ とする時、Triplet network における損失関数 (**triplet loss** と呼ばれます) は、次式で与えられます。

$$L = \max\left(D_p - D_n + m,\ 0\right)$$

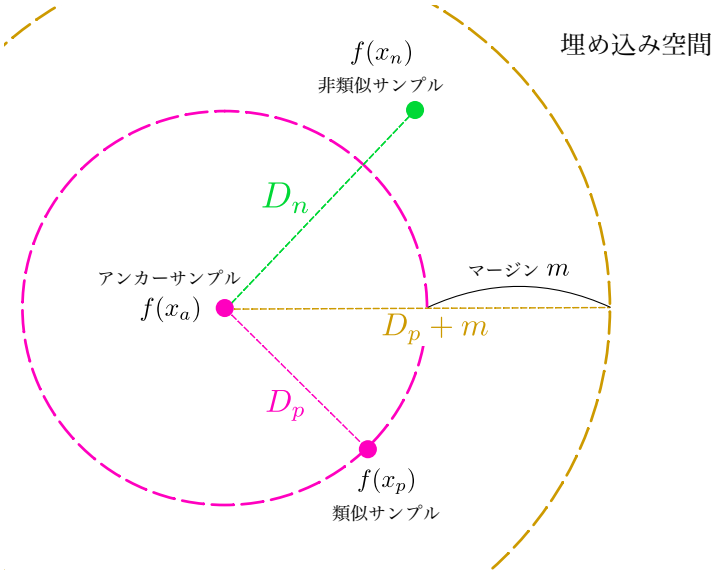
m は Siamese network 同様、マージンを表すハイパーパラメータです。また、D_p はアンカーサンプルと類似サンプルの特徴ベクトル間の距離、D_n はアンカーサンプルと非類似サンプルの特徴ベクトル間の距離を表しています。距離の測り方は Siamese network 同様、埋め込み空間の2つのベクトルの距離を計測できる関数であれば何でも良いのですが、ユークリッド距離を使うことが多くなっています。

$$D_p = \|f(x_p) - f(x_a)\|_2, \quad D_n = \|f(x_n) - f(x_a)\|_2.$$

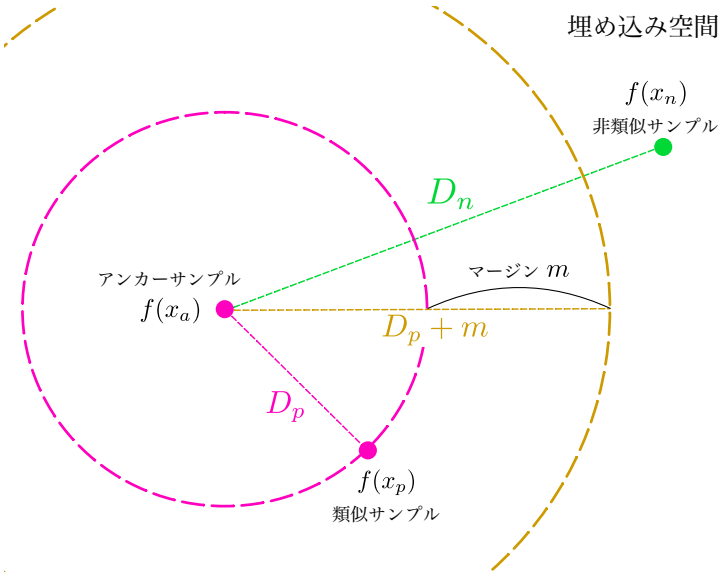
それぞれの文字が何を表すか確認したところで、triplet loss の意味を考えていきましょう。triplet loss には max 関数が含まれており、次のような場合分けが生じます。

$$L = \begin{cases} D_p - D_n + m & (D_p - D_n + m > 0 \ \Leftrightarrow\ D_p + m > D_n \ \text{の場合}) \\ 0 & (D_p - D_n + m \leq 0 \ \Leftrightarrow\ D_p + m \leq D_n \ \text{の場合}) \end{cases}$$

つまり、tripret loss では $D_p + m > D_n$ の場合に損失関数が正の値を持ってペナルティーが課される構造になっています。この $D_p + m > D_n$ という状況は、絵で表すと次のような状況です。



非類似サンプルが、アンカーサンプルから距離 $D_p + m$ 以下の領域に入っており、十分に非類似サンプルの位置を離すことが出来ていません。逆に、損失関数が 0 となる $D_p + m \geq D_n$ の場合は、次のような状況です。



非類似サンプルが、アンカーサンプルから距離 $D_p + m$ 以下の領域の外に出ており、十分に異なるサンプル同士が離れた状況になっています。このような理由から、tripret loss を最適化することで異なるクラスのデータが遠くに配置されるようにネットワークを学習することが可能になっています。

♣ Simese netwok と Triplet network の比較

Triplet netwprk の提案論文 [Hoffer+, arXiv:1412.6622 (2014)] では、Triplet netwprk は Siamese network の欠点を改良した手法であると主張されています。先ず、Siamese network で問題となった同じクラスのペアと異なるクラスのペアの間に生じる不均衡は、Triplet network では解消されています。Triplet network では同じクラスのデータの距離 D_p を 0 にする必要はありません。つまり D_p と D_n それぞれに対する条件をなくし、あくまで D_p と D_n を相対的に最適化することで、Siamese network で問題となった不均衡を解消しています。

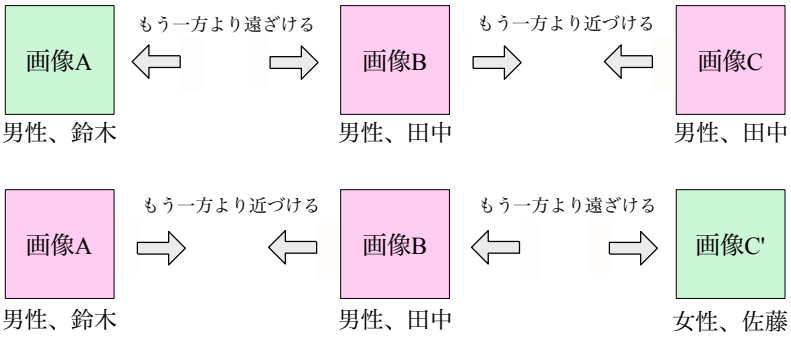
さらに、Triplet network には Siamese network と比べ、学習時のコンテキストに関する制約が緩和される利点があります。Siamese network で学習を行う際には、2 つのサンプルを近づけたいのか (類似サンプルとするか)、遠ざけたいのか (非類似サンプルとするか) 判断するには、コンテキストを考慮する必要があります。例として、性別と名前という 2 つのラベルを持ったデータセットを距離学習する場合を考えみましょう。この時、次のようなラベルを持った 2 つの画像 A・B があったとします。

- 画像 A：男性、鈴木
- 画像 B：男性、田中

この 2 つの画像は類似サンプルと見なして近づけるべきでしょうか？それとも非類似サンプルとして遠ざけるべきでしょうか？ これは画像のみからは判断できずタスクの内容 (コンテキスト) を考慮する必要があります。例えば、性別を識別するタスクであれば、画像 A・B はどちら

も男性であるため近づけるべきです。逆に顔認証のようなタスクでは、鈴木さんと田中さんは別人であるため画像 A・B は遠ざける必要があります。このような判断が Siamese network で学習を行う場合には必要となります。

一方、triplet loss の場合には、基準となる画像 (アンカーサンプル) に対して類似度が低いサンプルをもう一方のサンプルよりも遠ざけることになるため、コンテキストを考慮する必要はありません。



以上のことから、現在の深層距離学習では圧倒的に Triplet network (とその亜種) が利用されるようになっています。このように Triplet network は非常に優れたアルゴリズムですが、Siamese network にはなかったいくつかの問題を抱えています。その問題点と考案されている改良方法について、次のセクションで簡単に説明します。

♣ Triplet network の問題点

最後に、Triplet network の問題点について簡単に確認しておきましょう。

1. 学習がすぐに停滞してしまう

Triplet network の欠点の 1 つに、学習がすぐに停滞してしまう点が挙げられます。これは、学習データセットのサイズが増えてくると考えうる入力の組み合わせが膨大になり、かつその殆どの組み合わせが学習が進むにつれてパラメータ更新に影響を及ぼさなくなるからです。

入力の組み合わせが膨大になる点について、簡単な計算で確かめてみましょう。例として、クラスが 3 つ、1 クラスあたりのサンプル数が 3 つ (つまり全サンプル数は $3 \times 3 = 9$) というデータセットがあるとします。先ず、アンカーサンプルの選び方が 9 通りあります。そして同じクラスのサンプルの選び方は、アンカーサンプルを除いた $3 - 1 = 2$ 通りとなります。異なるクラスのサンプルは、全部で $9 - 3 = 6$ 通りあります。すると、Triplet network での入力の組み合わせは、全部で

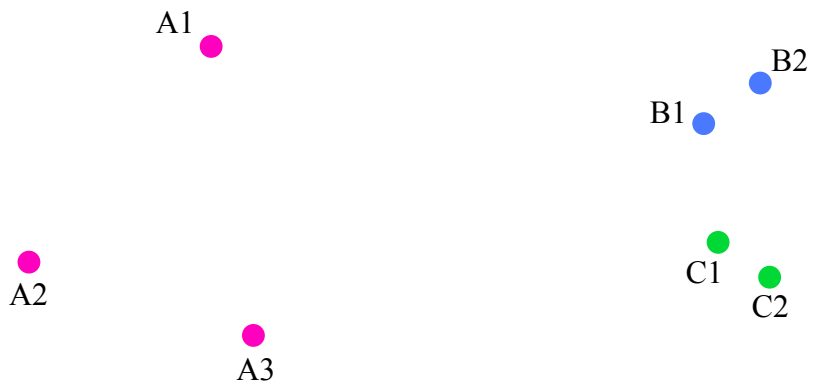
$$9 \times 2 \times 6 = 108 \text{ 通り}$$

となります。たったサンプル数が 9 のデータセットであってもこんだけ入力の組み合わせがあるため、学習データセットのサイズが増えると時間の都合で全ての組み合わせを入力することができなくなります。そのため、膨大な入力セットの組み合わせ (triplet) の内、学習に有効な

入力セットを厳選する必要があります。このような入力セットの厳選の操作のことを **triplet selection** や **triplet mining** と言います。学習に有効な入力セットとは、triplet loss が発生するようなできるだけ「難しい入力セット」のことです。一般に、学習が進むほど殆どのデータは適切に分類できるようになるため、学習スピードを上げるには難しい入力セットを選ぶ必要があるのです。多くの場合でこのような triplet selection が必要となるため、Triplet network の実装は煩雑になる傾向にあります。具体的な triplet selection の方法については本解説プリントでは触れませんが、近年の研究で様々な高速に triplet selection を行う手法が提案されているので、興味のある方は調べてみて下さい。

2. クラス内距離がクラス間距離より小さくなることを保証しない

Triplet network では繰り返し学習されることにより、可能な全ての入力セットに対して、$D_p < D_n$ が満たされるようになります。例えば、下図は 3 つのクラスからなる学習データに対して、Triplet network での最適化が完了した時の埋め込み空間の様子を表した例となっています。



アンカーサンプルを A3 とすると、類似サンプル A1 との距離 D_p は全ての非類似サンプルとの距離 D_n より短くなっており、$D_p < D_n$ が確かに満たされています。同じようにアンカーサンプルが C1 の場合でも、類似サンプル C2 との距離は全ての非類似サンプルとの距離より短くなっています。しかし上の状況では、クラス A のクラス内距離がクラス B とクラス C とのクラス間距離よりも大きいという状況が起きています。このように、triplet loss はクラス内距離がクラス間距離よりも小さくなることは保証しておらず、埋め込み空間が上図のような状況になってしまうケースがあります。

この問題を解決するために、入力セットを 3 つのデータから構成するのではなく、4 つのデータで構成する **Quadrupt loss** と呼ばれる次の損失関数が提案されています[10]。この Quadrupt loss の内容は本解説プリントの範囲を超えるため詳しい解説は省略しますが、4 つのデータを一組として学習を行うことで、任意のクラス間距離が任意のクラス内距離よりも必ず大きくなるようになります。

[10] arXiv:1704.01719, W. Chen+, "Beyond triplet loss: a deep quadruplet network for person re-identification" (2017)