

< A I についての考察 >

近年 A I という言葉が流行語のように言われるようになりました。

訳して人工知能(Artificial Intelligence)ですが、いまだ意識や感情をもつ状態にはいたっていません。しかし、量子コンピューターの発展と共に、早晚、人間的な人工知能が出現すると確信します。

コンピューターで考えさせるという意味では、第 2 次大戦時、イギリスのアラン・チューリングが開発した、暗号解読機エニグマが最初の A I だと私は考えます。

この頃、兵員、物資の輸送、効率の良い攻撃、防御などの戦略的な意味合いで、数理統計をベースにした、応用数学が発展し、特に、最適性を求める手法全体、オペレーションズリサーチ（O R）というジャンルがありました。

戦後、米国において資本主義経済の基本、利益追求において、利得を最大にする、企業戦略、株の取引等で、オペレーションズリサーチは開花しました。

同時に、多変量解析という分野も、発展しました。これも企業戦略で、分析や利益追求に使用されます。

これらも今では A I と呼べるものでした。

しかし、手法があきらかなるもので、プロセスもわかる決定論的なものでした。

最近の A I、深層学習（Deep Learning）は脳神経細胞をまねてのパーセプトロンが 1 9 5 0 年代で、それから発展して、2 0 1 0 年代にニューラルネットワークとして、進化しました。

これは、非決定論的なもので、途中プロセスはわかりません。答えが正解として出てくるかどうかも定かではないが、ガラガラポンとして出された答えに対して、学習機能を持たせることが出来長い目でみれば、頼もしいものと言えます。下図に私見ですが、表にしました。

そして、No 1 5 のニューラルネットワーク（深層学習）は No 1 - 1 4 の方法論を例外はありますが、包含しているのです。

とはいえ、それぞれのジャンルの A I 手法を、すべて深層学習（Deep Learning）手法で行うのは得策では無い。答えを出す時間（収束する時間）が遅くなり、深層が多層すぎて、メモリーオーバーになったりします。

A I は旧来の手法から、最新の手法まで、選択して、使用又は複数使用して比較したほうが効率が良いと考えます。下記、そのジャンルと大体の用途を示します。

非線形計画法（旧）という難解なジャンルがありますが省いています。私見ですが、1 5 のニューラルネットワークは閾値の判定にジグモイド関数という非線形な関数を使用するので、代替えがきくのではないかと考えます。

No	新旧別	A I ジャンル	内容
1	旧	線型計画法（O R）	最大利得条件を求める
2	旧	スケジューリングアルゴリズム（O R）	P E R T、C P M等の工程スケジュール、巡回セールスマン最短コース問題

3	旧	モンテカルロ法（OR）	乱数を使って、シミュレーション構造、様式の分析に使用
4	旧	待ち行列、待ち合わせ（OR）	窓口サービス業務、交通渋滞
5	旧	ゲームの理論（OR）	ミニマックス法、ベイズ戦略、最大マッチング理論
6	旧	回帰分析（多変量解析）	単、重回帰方程式にて目的変量の分析
7	旧	主成分分析（多変量解析）	複数の変量から、主成分となる変量を割り出す
8	旧	因子分析、構造方程式モデル（SEM）（多変量解析）	資料から、原因になるものを割り出す
9	旧	判別分析（多変量解析）	マハラノビスの距離を使っの判別は私は多用しています
10	旧	自己相関、相互相関	パターン認識
11	旧	ファジー推論	認識
12	中古	最尤推定、ベイズ推定、EMアルゴリズム、	教師なしパターン認識
13	中古	マルコフ過程（3とMixでMCMC法）	状態遷移確率、時系列上の予測
14	中	サポートベクトルマシン（SVM）	多変量解析の判別分析のジャンルで、クラスターに分離、k-平均法等がある
15	新	ニューラルネットワーク	AIの横綱、深層学習の基本となる

過去の経験で、大手鉄鋼メーカーのL型鋼の疵分布を削り取るスケジューラーを、2の巡回セールスマン最短コース問題と5の最大マッチング理論の混合型で解決してます。

後日、年間数億円の利益を生み出した記事が日刊工業新聞に出ていました。35年前ぐらいで当時著作権など、うるさくない時代で、ソフト数百万円で約半年開発して終わりでした。

上段、下段に左右合計4個の研磨機が固定された、台車をいかに最短で、疵を削るかという問題で、台車の幅×2でくぎられた、領域同志の、最大マッチング理論上、もっとも効率の良い、ずらし方カプリング（MAX）で、その中はトポロジーの一筆書きのように、巡回セールスマン最短コース問題で、処理する（MIN）というものです。天文学的な組み合わせ状態サーチをさせて、100点満点にはならないが、高得点の成果を出したのです。天文学的な組み合わせ状態サーチをやれば、ベストな解はひとつ見つかりますが、その演算時間は、いかほどか。量子コンピューターならば問題ないと思いますが、そうでなければ、AIが必要になります。

ちなみに最大マッチング理論は別名婚活の理論とも言われ、見合い会場で、男女のそれぞれの相手に対する評価点から、個人の満足度ではなく、主催者側のカップル成立数を最大にする組み合わせ決定なのです。

＜A I による物識別処理＞

最近、A Iによる処理が多岐にわたって、行われています。科学、医療、経済、商業と大きく貢献しつつあります。細かく言うと、工業生産技術、測定検査、医療検診検査、株価予測、マーケットリサーチ、企業診断、マクロ経済分析、娯楽ゲーム等、私にも知らないジャンルがあるかと思います。

今回、物の識別を対象としたDeep Learningを紹介いたしますが、入力の実体性は何でもよく、抽象的な事象でもかまいません。

文字の識別や顔認証などの画像処理、音声識別など、直接A I処理する方法は、入力数が多いので、メモリーとその処理時間の問題が、不明なため避けています。

Deep Learning を使うには、前段、画像処理を完成させ、Deep Learning入りに連結させる方法が、メモリーとその処理時間の短縮につながり、学習時間も短くすみます。

Deep Learningを完成させる要因として、出力が何かという事と、入力をどうするのかという事をよく考え、吟味する事から始めます。入力の種類と性質、その数を決めます。その際、入力同志は独立のもので、従属するものは適切ではありません。

Deep Learning入力CH間は一般的に同一座標である必要はありませんし、線型な入力である必要もないのですが、比較データのCH間は同一座標系にして、非線形の場合は、線型に変換して入力するほうが、より確実な成果を得ると考えます。数学的な証明しているわけではないのですが、出来るだけきれいな形にして、入力したほうがいいと思います。数学的な表現では正規直交基底のベクトル空間座標を構築します。

次に教師データをどうするかです。

Deep Learningでは、教師データを学習させますが、その方法を検討すべきです。

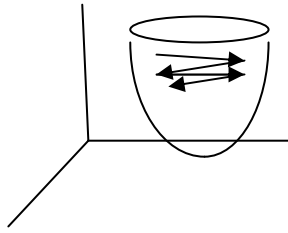
たくさんの教師データを学習させなければなりません。特に中間層が多い場合は、相当な量の学習をしなければなりません。Deep Learningでは過学習という問題はなさそうです。

取り扱うモードとして、学習、運用、自動 とあります。

自動の場合、通常画像処理2方式で、一致したものだけを教師として、学習させて、Deep Learningの結果と一押しの通常画像処理との誤差率が一定値を下回ったらDeep Learningに切り変える。誤差率が一定値を上回ったら一押しの通常画像処理に切りかえるというDeep Learningとしては保険のかかった安全なものにします。

しかし、ノイズの多い悪条件では、通常画像処理自体の誤差が少なくならないので、前もって、決められた教師データ列を、ティーチングモードで、教え込む方法もあります。

その場合、例えばOKのものとNGのもののような異種のものを交互、又はN回に1回変えるように入力し、同種のものを続けて学習させてはいけません。下図のように、徐々に底に収束させるイメージです。



弊社Deep LearningはC++で製作しています。海外メーカーのツール等使用しません。初期処理関数と終了関数以外は、関数一つで、実行又は学習を行うもので、非常にシンプルで、使いやすいものです。具体例として下記のようなものです。

初期処理は省きます。

Deep Learningの関数は以下の形式です。

```
int AI_DeapLearning(double* INData, int INTeach, int ActSet, double NoTHVAL, int SaveCount, double Reliable, double* Jisseki, int* Mmode, double* EOUT)
```

使い方は

```
int GetTriggerKindFromDeepLearning(int Gano)
{
    int64 red = mMahaRate->GetSumInt64(MahaResult::RED_RES);
    int64 yellow = mMahaRate->GetSumInt64(MahaResult::YELLOW_RES);
    int64 black = mMahaRate->GetSumInt64(MahaResult::BLACK_RES);
    int64 Binned = mMahaBin->GetSumInt64(MahaResult::BIN_RED_RES);
    int64 Bingreen = mMahaBin->GetSumInt64(MahaResult::BIN_GREEN_RES);
    int64 Binblue = mMahaBin->GetSumInt64(MahaResult::BIN_BLUE_RES);
    double Jigen;
    double FeretRatio;
    int iLbLB = mMahaBin->GetMaxFeret(MahaResult::LbL_B);
    double INData[9];    //9種の入力
    int TeachBinRate;    //学習 0 : 対象外、 1 : もの 1、 2 : もの 2、 3 : もの 3
    int ActSet;    //0:学習指示  1 : DeepLearning 運用指示  2 : 自動指示
    double NoVal = 0.2;    //この数値以下は対象外となる。即ち、出力全部 0
    int SaveCount = 100; //更新重み係数セーブ回数割合（100回に1回セーブ）
    //double Target = 90.0; //自動指示の時の、目標信頼比率 %（高くすると永遠に達しないことがある。その時はずっと学習モード）
    double Target = GetDeepLearningTrr();
    double Jisseki;    ///実績信頼比率 %
    int Mode;    //0:学習中  1 : DeepLearning 運用中
    double Eout[3];    //3種の区別出力
    int res_ai = 0;
```

```
INData[0] = (double)yellow / 100000;  
INData[1] = (double)red / 100000;  
INData[2] = (double)black / 100000;  
INData[3] = (double)Binred / 1000;  
INData[4] = (double)Bingreen / 1000;  
INData[5] = (double)Binblue / 1000;  
INData[6] = Jigen / 10000;  
INData[7] = FeretRatio;  
INData[8] = (double)iLbIB / 100;
```

```
res_ai = AI_DeapLearning(INData, TeachBinRate, ActSet, NoVal, SaveCount, Target, &Jisseki, &Mode,  
Eout);  
SetDeepLearingJisseki(Jisseki);  
SetDeepLearingMode(Mode);  
SetDeepLearingEout0(Eout[0]);  
SetDeepLearingEout1(Eout[1]);  
SetDeepLearingEout2(Eout[2]);  
return res_ai;
```

```
}
```

です。

以上 (有) 加藤エムイーピーシステム