ＲＴＭによるコンポーネント化作業
「学習・推論コンポーネント」
ユーザーマニュアル

平成 19 年 05 月 31 日

株式会社アドイン研究所

AdIn Research, Inc.

【目次】

１．はじめに

本書は、RT ミドルウェア上で動作する RT コンポーネント「学習・推論コンポーネント」に関して、その機能、使用手順、仕様等を説明するものである。

2. 学習・推論コンーネントの機能概要

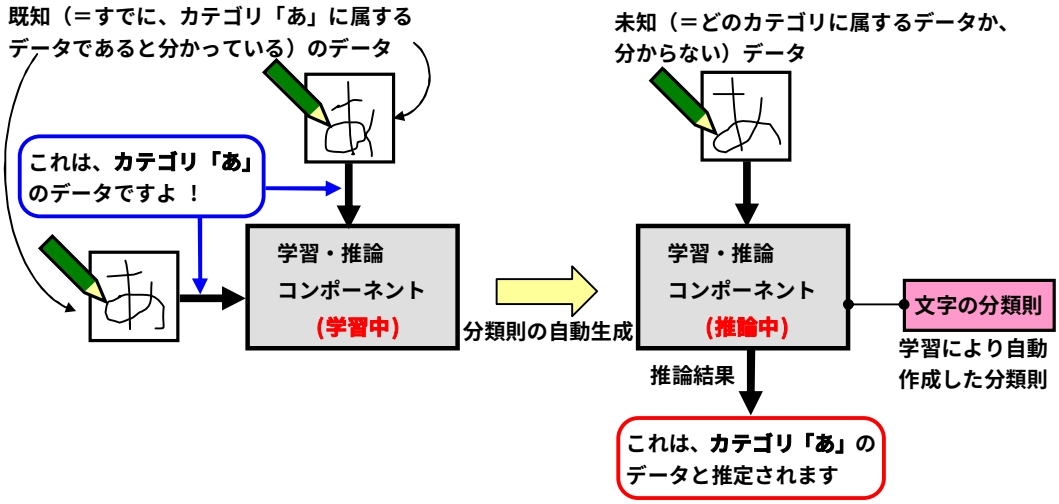
学習・推論コンポーネントは、株式会社アドイン研究所(*1)のファジィニューロ学習・推論エンジンであるβ-RNA(*2)を、RT ミドルウェア上で動作するように、RT コンポーネント化したものである。本章では、学習・推論コンポーネントの機能について概説する。

2.1. 概要

学習・推論コンポーネントは、学習と推論を行なうコンポーネントである。ここでいう“学習”とは、人工知能研究の一分野である機械学習のことであり、サンプルデータ集合を対象に解析を行い、有用な規則やルールを抽出することである。また“推論”とは、”学習”の結果、抽出したルールをもとに、未知のデータに対して、新しい結論を得ることである。学習・推論コンポーネントが行う、学習・推論の具体的な処理は、以下の通りである。

学習： 既知のデータ（＝カテゴリ分類されているデータ）を統計的に解析して、カテゴリ分類則を、生成する。
推論： 生成したカテゴリ分類則を用いて、未知のデータに対して、カテゴリ分類を行う。

学習・推論コンポーネントは、未知のデータに対して推論を行い、結果（そのデータのカテゴリ）を得ることを目的とする。つまり、学習・推論コンポーネントは分類器として動作する。学習・推論コンポーネントの特徴は、学習により、分類則を自動生成するという点であり、これにより、分類則をプログラムとして記述するのが難しい分野（文字認識、画像認識等のパターン認識の分野など）に対しても、カテゴリ分類を行うことができる。下図に、学習・推論コンポーネントを使用して、文字認識を行う例を示す。



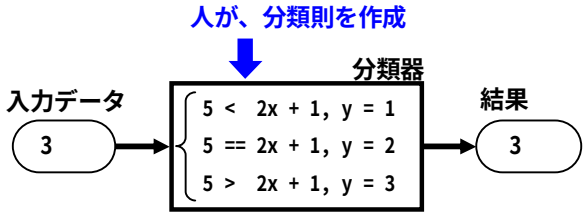
(*1) http://www.adin.co.jp/index.html

(*2) http://www.adin.co.jp/products/brna/brnatop.htm

２.２. 学習機能

データをカテゴリに分類するシステムを作成する場合、通常は、分類則を人が作成する。つまり、分類則の記述形式は、プログラムとして記述可能な、数式（下図参照）、あるいは、IF～ELSE～THEN 形式（下図参照）に限られる。

数式で記述されたルールの例

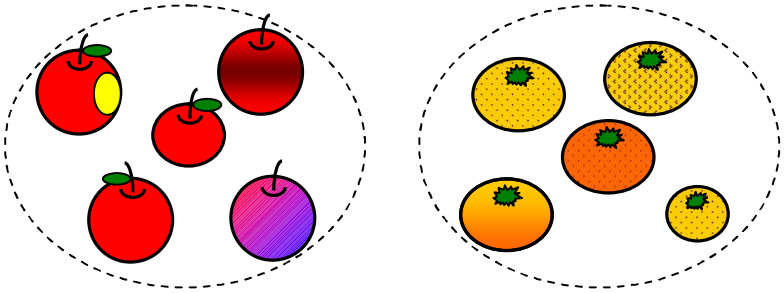


IF～ELSE～THEN 形式で記述されたルールの例

ところで、現実世界において、人は、あいまいさを持った対象でも、きちんと分類することができる。例えば、人は、「りんご」と「みかん」とを分類する場合、少々青みがかった「りんご」であっても、真っ赤に熟した「りんご」であっても、「りんご」と判定できる。それにもかかわらず、「りんご」と「みかん」との分類則の作成は、あいまいさを記述する必要がある（＝数式、IF～ELSE～THEN 形式での記述が難しい）ため、困難である。

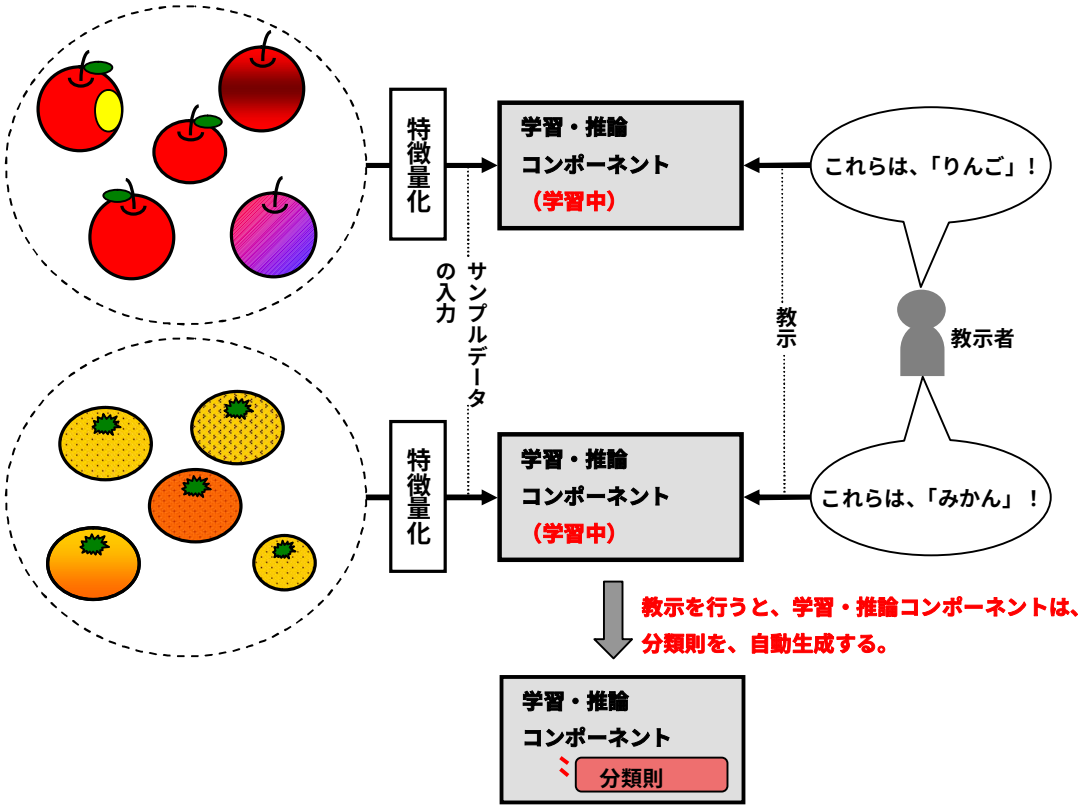
このような場合、学習・推論コンポーネントの学習機能を用いることで、分類則を自動生成できる。以下に、学習機能を用いて、「りんご」と「みかん」の分類則を自動生成する手順を述べる。

① カテゴリ「りんご」に属するサンプルデータと、カテゴリ「みかん」に属するサンプルデータを用意する。



りんごや、みかんそのもの（果物）を、学習・推論コンポーネントに直接入力することはできないため、入力用のデータとして、りんごや、みかんの特徴を現すデータを用意する必要がある。例えば、りんごやみかんの画像を撮影して、その各ピクセルの RGB 値の集合を、入力用データとする。なお、このように、特徴を現すデータを作成することを、**特徴量化**とよぶ。

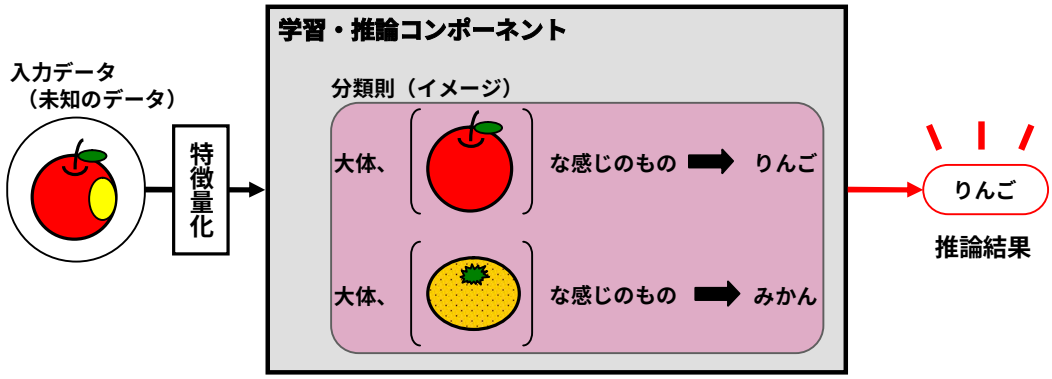
② サンプルデータを特徴量化して、学習・推論コンポーネントに入力する。それと同時に、そのサンプルデータが、カテゴリ「りんご」に属するものか、カテゴリ「みかん」に属するものかを、学習・推論コンポーネントに入力する（これを、**教示**という）。



③　上記の操作を行うと、学習・推論コンポーネントは、分類則を自動生成する。このように、分類則を自動生成するためは、特徴量化と、教示が必要になる。

２.３. 推論機能

前章で、学習・推論コンポーネントの学習機能を説明した。学習機能により生成された分類則を用いることで、未知のデータに対しても、「それが、どのカテゴリに属するのか？」ということを判定することができる。例えば、前章では、「りんご」と「みかん」の分類則を自動生成したが、この分類則を用いた場合、未知のりんごのデータ（教示に用いたサンプルデータ以外のデータ）を特徴量化して、学習・推論コンポーネントに入力すると、それが、カテゴリ「りんご」に属するデータであると、推論結果が出力されることになる。



３．学習・推論コンポーネントの使用手順

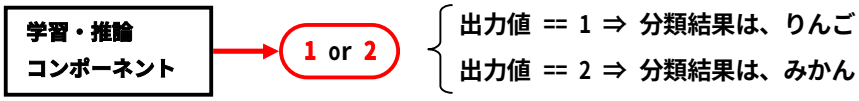
本章では、学習・推論コンポーネントの使用手順を、カラー画像による、果物の分類システムを例にして、説明する。

３.１．分類対象のカテゴリ設計

はじめに、学習・推論コンポーネントに出力させたいカテゴリを決定する。学習・推論コンポーネントは、カテゴリを、ID（0 以上の整数）として扱うため、分類対象のカテゴリを決定したら、各カテゴリに対して、ID を割り当てる。果物分類システムでは、果物の中で、りんごとみかんのみを分類対象とすることにして、下表のようなカテゴリ設計を行う。

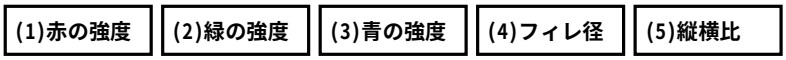
カテゴリ	ID
りんご	1
みかん	2

これにより、果物分類システムの学習・推論コンポーネントは、下図のような出力を行うことになる。

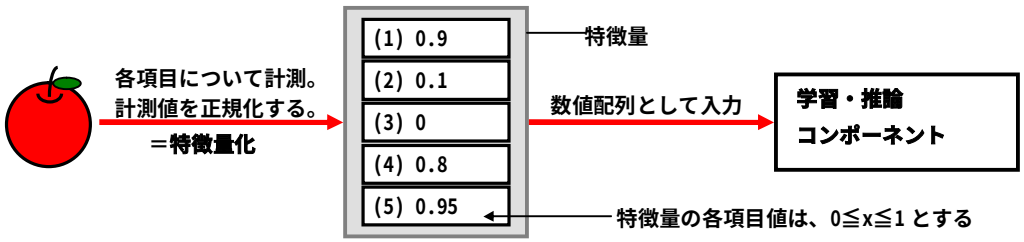


３.２．分類対象の特徴量化の設計

学習・推論コンポーネントは、分類対象を、0～1 の数値配列として扱う。従って、分類対象を、0～1 の数値配列で表現する必要がある。このため、まず、分類対象を、どのような項目で表現するかを設計する。果物分類システムでは、分類対象を、以下の 5 項目で表現することにする。



次に、分類対象を、各項目について計測して、計測値を正規化する（= 0 以上 1 以下の数値に変換する）。以下に、果物分類システムで、りんごの画像を学習・推論コンポーネントに入力するときの例を示す。



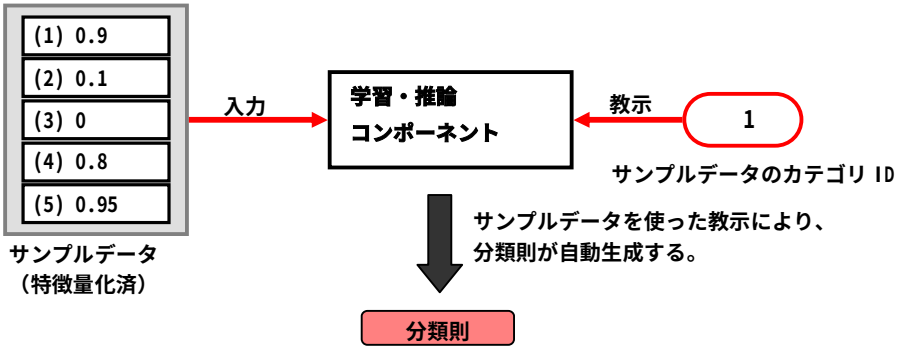
３.３. 設定

学習・推論コンポーネントを、学習モード、推論モードで動作させるための準備として、学習・推論コンポーネントに、以下の情報を与える。

1. 特徴量の項目数（果物分類システムの場合、'５'）
2. 学習結果等を保存するための、ディレクトリのパス（詳細は、『4.1. 章データファイル』を参照）

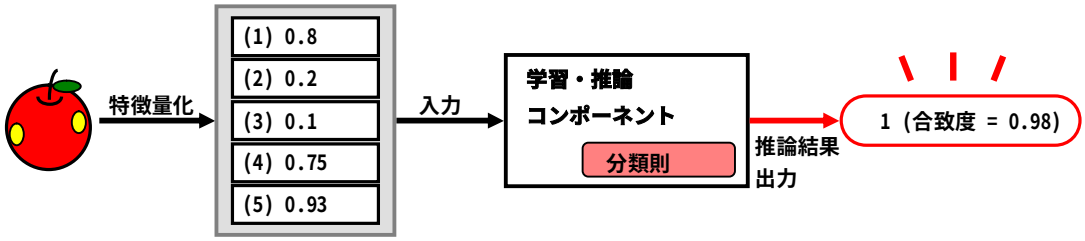
３.４. 教示

「分類対象のカテゴリ設計」、「分類対象の特徴量化の設計」、「設定」が終了したら、学習・推論コンポーネントを、学習モードで動作させる。学習モードの学習・推論コンポーネントに教示を行うと、学習・推論コンポーネントは、分類則を自動的に生成する。具体的には、特徴量化されたサンプルデータと、それが属するカテゴリの ID とを、学習・推論コンポーネントに与えることで、教示を行う。



３.５. 推論

「教示」が終了したら、推論のための準備が完了したことになる。従って、学習・推論コンポーネントを、推論モードで動作させる。推論モードの学習・推論コンポーネントに、新しい（未知の）データを特徴量化して、入力すると、学習・推論コンポーネントは、「入力データがどのカテゴリに属するか？」を推論して、推論結果を ID で出力する。



なお、学習・推論コンポーネントは、推論結果として、ID と併せて、**合致度**（０以上１以下の数値）を出力する。合致度は、「推論結果が、どの程度信頼できるか？」ということを表す指標である。

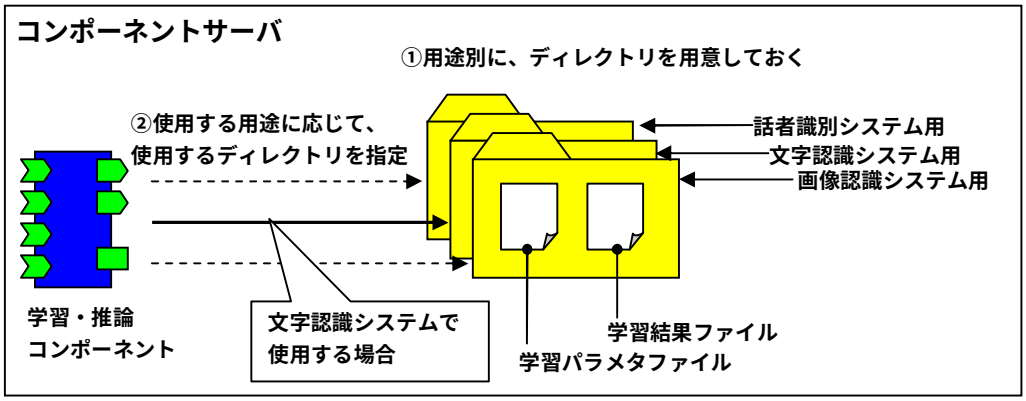
４．学習・推論コンポーネントの仕様

４.１．データファイル

学習・推論コンポーネントは、２つのデータファイルを使用する。詳細を、下表に示す。

名称	ファイル名称	説明
学習パラメータファイル	rs1.ila	学習・推論コンポーネントの設定ファイル。学習・推論時に使用するパラメータが保存される。通常は、編集する必要はない。学習・推論コンポーネントのチューニングを特別に行う場合は、β -RNA の仕様書『rs1 データファイル書式仕様書.pdf』を別途参照のこと。
学習結果ファイル	rs1.bra	学習機能で学習した結果が保存される。学習時にファイルが存在しない場合は、最初の学習が終了した段階で、ファイルが自動生成される。既にファイルが存在した場合は、学習結果が、追記（上書きではない）される（＝追加学習が行われる）。

学習パラメータファイル、学習結果ファイルは、学習・推論コンポーネントの用途ごとに用意する必要がある（例えば、話者識別用の学習パラメータファイル / 学習結果ファイル、画像認識用の学習パラメータファイル / 学習結果ファイルなど）。学習パラメータファイル、学習結果ファイルは、一つのディレクトリ内に、各ファイル一つずつしか作成できないため、用途ごとに、ディレクトリを用意しておく必要がある。



学習・推論コンポーネントに学習・推論処理を行わせる前に、用途に応じて、使用するディレクトリを指定する。なお、学習パラメータファイル、学習結果ファイルは、コンポーネントサーバマシン（クライアントマシンではない）に存在する必要がある。

４.２. コンポーネントの形態

RT コンポーネントには、２通りの利用形態が定義されている。一つは、データポート（InPort/OutPort）を利用する形態であり、もう一つは、サービスポートを利用する形態である。

学習・推論コンポーネントは、これらの形態に対応した、２種類のコンポーネントが用意されている。即ち、データポートを利用する形態の学習・推論コンポーネントと、サービスポートを利用する形態の学習・推論コンポーネントである。前者のコンポーネント名称は、BetaRnaPort であり、後者のコンポーネント名称は、BetaRnaService である。

コンポーネント名称	説明
BetaRnaPort	データポートを使用する形態の、学習・推論コンポーネント
BetaRnaService	サービスポートを使用する形態の、学習・推論コンポーネント

４.３. プロファイル

学習・推論コンポーネントのプロファイルを、以下に示す。

プロファイル項目		値
モジュール名	(RTC_MODULE_NAME)	BetaRnaPort (データポート版) BetaRnaService (サービスポート版)
概要	(RTC_MODULE_DESC)	Beta Rna in/outpurt component (データポート版) Beta Rna serviceport component (サービスポート版)
バージョン	(RTC_MODULE_VERSION)	0.1
作成者	(RTC_MODULE_AUTHOR)	-
カテゴリ	(RTC_MODULE_CATEGORY)	Generic
コンポーネント型	(RTC_MODULE_COMP_TYPE)	COMMUTATIVE
アクティビティ型	(RTC_MODULE_ACT_TYPE)	SPORADIC
最大インスタンス数	(RTC_MODULE_MAX_INST)	10
モジュール記述言語名	(RTC_MODULE_LAN)	C++
モジュール記述言語型	(RTC_MODULE_LANG_TYPE)	compile

４.４. インターフェース

学習・推論コンポーネントのインターフェースの仕様を、以下に示す。

４.４.１. BetaRnaPort のインターフェース

データポート（InPort/OutPort）を使用する形態のコンポーネント BetaRnaPort の、データポートの仕様を下表に示す。

入力ポート（InPort）仕様

ポート名称	データ型	説明
ID	TimedShort	動作モード（学習モード/推論モード）の切り替えを行う。学習モード時は、教示のため、サンプルデータのカテゴリを指定する役割も併せ持つ。 (学習モード時) サンプルデータのカテゴリ ID (推論モード時) -1
ParamDir	TimedString	データディレクトリ（学習パラメタファイル、学習結果ファイルを格納するディレクトリ）を指定する。なお、データディレクトリは、コンポーネントサーバに存在する必要がある。
DataCount	TimedShort	学習モード時はサンプルデータの、推論モード時は、分類対象データの、特徴量の項目数を指定する。
DataAry	TimedDoubleSeq	学習モード時はサンプルデータの、推論モード時は、分類対象データの、特徴量の配列を指定する。各特徴量は、0 以上 1 以下でなければならない。

出力ポート（OutPort）仕様

ポート名称	データ型	説明
Result	TimedShort	推論モード時は、推論結果のカテゴリ ID が出力される。（学習モード時は、入力ポート ID で指定した、サンプルデータのカテゴリ ID が、そのまま出力される。）
Trusty	TimedDouble	推論モード時のみ合致度（推論結果が信頼できる度合い、0 以上 1 以下）が出力される（学習モード時は、 -1 が出力される）。

４.４.２. BetaRnaService のインターフェース

サービスポートを使用する形態のコンポーネント BetaRnaService の、インターフェース仕様を示す。BetaRnaService のインターフェースは、関数の宣言となる。

(1) 初期化関数

```
CORBA::Short
BetaRnaSVC_impl::Init(
    const char*    sRnaDir,
    CORBA::Short   iInputCh  )
```

機能：

学習・推論コンポーネントの初期化を行う。

引数：

sRnaDir：(in)

データディレクトリ（学習パラメータ、及び学習結果ファイルを保存するディレクトリ）を指定する。データディレクトリは、コンポーネントサーバに存在する必要がある。

iInputCh：(in)

学習モードで用いるサンプルデータの、あるいは、推論モードで用いる分類対象データの、特徴量の項目数を指定する。

戻り値：

成功時は、ゼロ、エラーの場合、非ゼロが返る。

注意：

コンポーネント動作中に、データディレクトリを変更する目的で、本関数を、複数回実行しても良い。しかし、このとき、iInputCh の値を変更してはならない。これは、特徴量の項目数は、コンポーネント動作中は、常に同一の値としなければならないという、学習・推論コンポーネントの制約によるものである（最初に設定した数値と異なる数値を設定にすると、エラーとなる）。

(2) 学習・推論関数

```
CORBA::Short
BetaRnaSVC_impl::Exec(
    CORBA::Short              iID,
    const BetaRna::DArray&    aInputData,
    CORBA::Short&             iResultID,
    CORBA::Double&            dTrusty   )
```

機能：

学習・推論を行う。

引数：

iID：(in)

本引数で、動作モード（学習モード / 推論モード）の切り替えを行う。学習モード時は、教示のために、サンプルデータのグループを指定する役割も併せ持つ。

(学習モード) サンプルデータのグループ ID

(推論モード) -1

aInputData：(in)

学習モード時は、サンプルデータの、推論モード時は、識別対象データの、特徴量の配列を指定する。各特徴量は、0 以上 1 以下でなければならない。なお、配列の要素数は Init() 関数の引数 iInputCh で指定した値と同じでなければならない。

iResultID：(out)

推論モード時には、推論結果のグループの ID を返す。

学習モード時には、iTalkerID で指定した値を、そのまま返す。

dTrusty：(out)

推論モード時には、合致度を返す。

学習モード時には、-1 を返す。

戻り値：

成功時は、ゼロ、エラーの場合、非ゼロが返る。

５．学習・推論コンポーネントの使用方法

本章では、学習・推論コンポーネントの使用方法を説明する。

５.１．ビルド方法

学習・推論コンポーネントのソースファイルをビルドする手順を、以下に示す。

(1) 開発フォルダ（src/RT フォルダ）に移動する。（cd src/rt）

(2) ビルド用のシェルスクリプト（build.sh）を実行する。

(3) 開発フォルダ直下の、実行フォルダ（run フォルダ）に、学習・推論コンポーネント関連ファイルが生成される。下表に、各ファイルの説明をまとめる。

コンポーネント名称	ファイル名称	説明
BetaRnaPort	BetaRnaPort.so	データポート版学習・推論コンポーネント本体
	BetaRnaPortComp	データポート版学習・推論コンポーネントを起動するために必要な実行ファイル
BetaRnaService	BetaRnaService.so	サービスポート版学習・推論コンポーネント本体
	BetaRnaServiceComp	サービスポート版学習・推論コンポーネントを起動するために必要な実行ファイル

５.２．実行方法

学習・推論コンポーネントを実行する手順を、以下に示す。

(1) 実行フォルダ（src/RT/run フォルダ）に移動する。（cd src/RT/run）

(2) RT コンポーネント設定ファイル rtc.conf の設定を行う。設定項目 corba.nameservers で、学習・推論コンポーネントを登録するネームサーバを指定すればよい。

(rtc.conf)

```
corba.nameservers: localhost.localdomain:9876
naming_formats: %n.rtc
logger.log_level: TRACE
```

ネームサーバアドレス　　ネームサーバポート番号

(3) 学習・推論コンポーネント起動スクリプト（start.sh）の設定項目 nsport に、rtc.conf の設定項目 corba.nameservers で指定したポート番号を設定して、start.sh を実行する。データポート版及び、サービスポート版の学習・推論コンポーネントが、起動する。

(start.sh の抜粋)

```
... (略) ...
nsport='9876'
hostname=`hostname`
....
```

ネームサーバポート番号

５.３. 終了方法

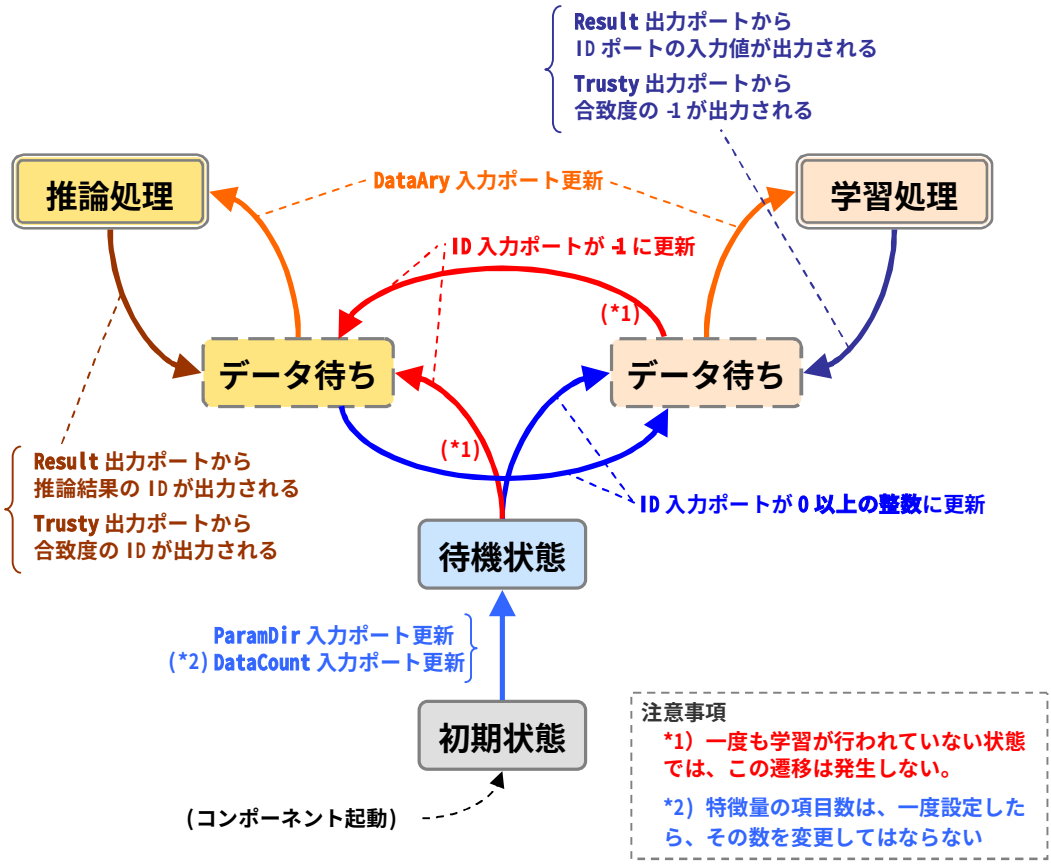
学習・推論コンポーネントを実行する手順を、以下に示す。

(1) 実行フォルダ（src/RT/run フォルダ）に移動する。（cd src/RT/run）

(2) 学習・推論コンポーネント終了スクリプト（stop.sh）を実行する。

５.４. BetaRnaPort の使用方法

BetaRnaPort は、データポートを、他の RT コンポーネントのデータポートと接続して使用する。下図に、データポートのデータ入出力と、BetaRnaPort の状態遷移との関連を示す。BetaRnaPort に学習処理、推論処理を実行させるには、本遷移図に従って、BetaRnaPort のデータポートにデータを入力すればよい。



なお、BetaRnaPort は、不活性化（Deactivate）により、初期状態に戻ることはない（再び活性化した場合、不活性化される直前の状態のままとなる）。

５.５. BetaRnaService の使用方法

BetaRnaService は、他の RT コンポーネントのサービスポート（サービス受付側のポート）と接続して使用する。接続により、他の RT コンポーネントは、サービスポート経由で、BetaRnaService が提供する関数を実行できるようになる。

サービスポート経由で、BetaRnaService を使用する RT コンポーネントを開発するには、BetaRnaService のインターフェース定義ファイル（IDL ファイル）が必要になる。以下に、IDL ファイルを示す。

IDL ファイル（ファイル名称：BetaRna.idl）

```
interface BetaRna
{
    typedef sequence<double>  DArray;

    short  Init(  in   string   sRnaDir
                  in   short    iInputCh   );

    short  Exec(  in   short    iID,
                  in   DArray   aInputData
                  out  short    iResultID
                  out  double   dTrysty    );
};
```

次に、IDL ファイルの使い方を示す。まず、「BetaRna.idl」ファイルを作成して、RT コンポーネントの開発ディレクトリ（ソースファイル等を置くディレクトリ）に配置する。次に、コードジェネレータ rtc-template のオプションで、「BetaRna.idl」ファイル及び、サービスポートの型、変数名称を指定する。指定方法を、以下に示す。

```
rct-template  -bcxx \
     --module-name="SampleComponent" \
     --module-type='' \
     ....
     --consumer=BetaRna:BetaRna0:BetaRna
     --consumer-idl=BetaRna.idl \
```

RT コンポーネントの、コンポーネントクラスに、m_BetaRna0 という名称のメンバ変数が作成される。

上記の rtc-template を実行（RT コンポーネントの開発ディレクトリで実行）すると、RT コンポーネントのコンポーネントクラスファイルが作成される。このコンポーネントクラスには、m_BetaRna0 という名称のメンバ変数が作成されている。このメンバ変数を使用して、BetaRnaService の関数を呼び出すことができる。以下に、関数呼び出しのプログラムの例を示す。

```
const char* pszMyDir = "setting";
int iInputCh = 10;
try {
  int iRet = m_BetaRna0->Init( pszMyDir, iInputCh );
}
catch(...) {
  // error
}
```

変数 m_BetaRna0 は、BetaRnaService を参照するポインタ変数として扱えばよい。ななお、BetaRnaService の関数を呼び出す処理は、必ず、例外処理（try～catch 構文）を記述する必要がある。これは、BetaRnaService と未接続の状態で、RT コンポーネントを実行して、BetaRnaService の関数を呼び出そうとすると、例外が発生するためである。

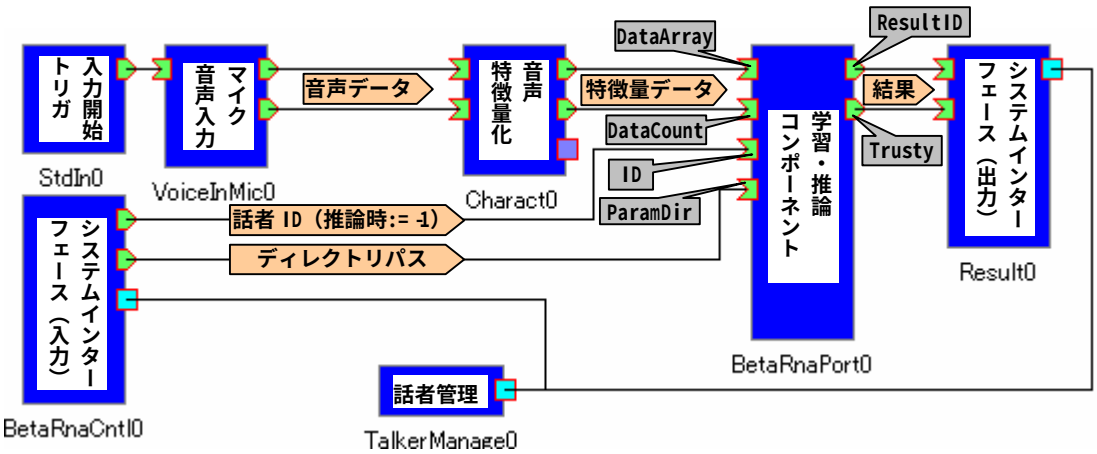
５.６. 注意事項

学習・推論コンポーネント（BetaRnaPort、BetaRnaService）は、学習時の条件と、追加学習、または推論時の条件が異なると、正常な学習 / 推論処理を行うことができない。ここで言う条件とは以下の２つである。

- 学習パラメータファイルの内容
- 入力される特徴量の、項目数

６. システム例

以下に、学習・推論コンポーネントを使ったシステムの例として、学習・推論コンポーネントを、話者識別システムに組み込んだときの、RT コンポーネントの接続図を示す。



７．動作環境

動作対象環境

- gcc バージョン 3.3.2 以上がインストールされている Linux(x86)環境
- RT ミドルウェア「OpenRTM-aist-0.4.0」

動作確認済み環境

- Vine-Linux 3.2
- RT ミドルウェア「OpenRTM-aist-0.4.0」

開発環境

- Vine-Linux 3.2 （gcc バージョン 3.3.2 でコンパイル）
- RT ミドルウェア「OpenRTM-aist-0.4.0」

８．その他

８.１．ライセンス等

学習・推論コンポーネントの著作権は、β-RNA ライブラリを除き、独立行政法人産業技術総合研究所に帰属する。

β-RNA ライブラリの著作権は、株式会社アドイン研究所に帰属する。

β-RNA ライブラリは、株式会社アドイン研究所の商品であるが、非商用利用に限っては保守・サポートなしを条件に無償で使用してよい。

８.２．連絡先

株式会社 アドイン研究所
〒102-0094 東京都千代田区紀尾井町 3-6 秀和紀尾井町パークビル 1F
TEL: 03-3511-2215　　FAX: 03-3511-3078
E-mail: rt_brna@adin.co.jp
URL: http://www.adin.co.jp

－以上－