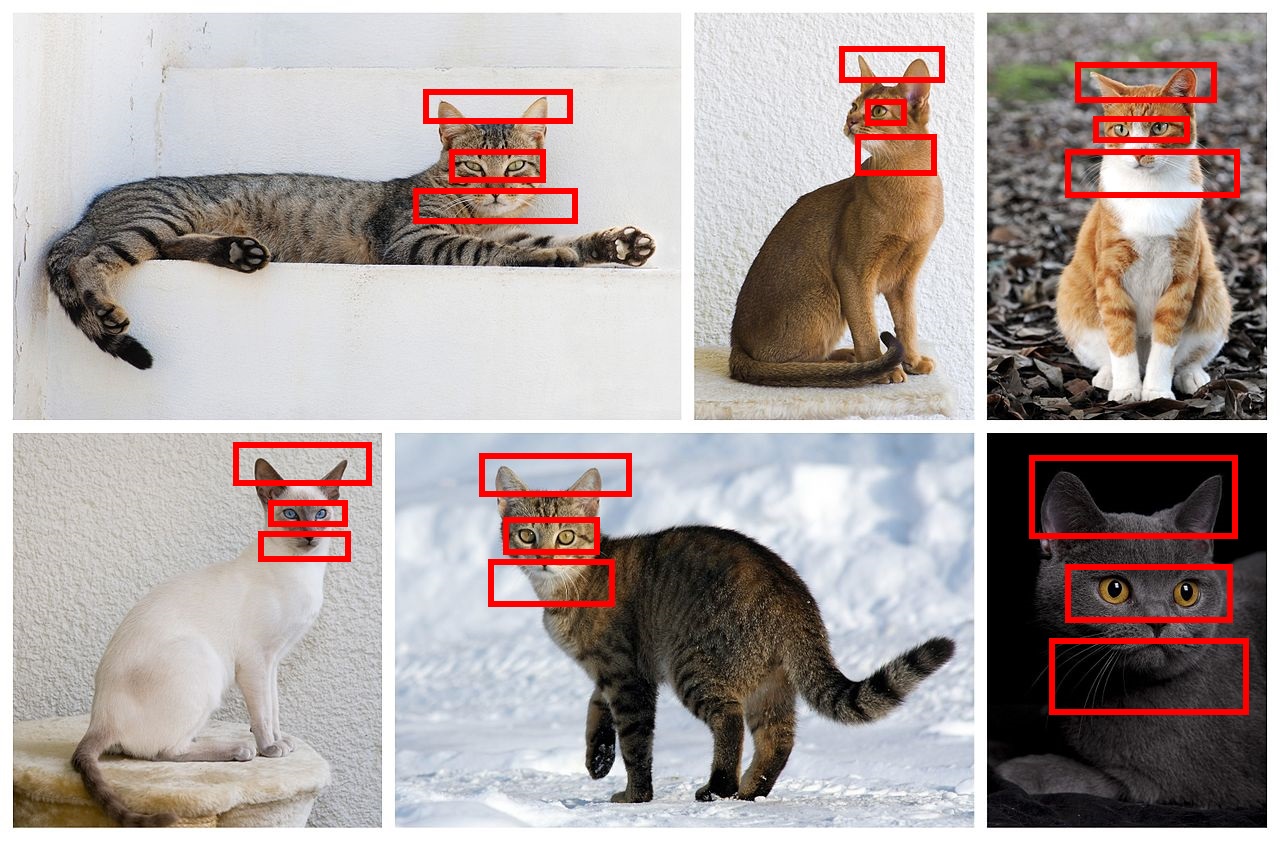
**Microsoft Cognitive Toolkit を使ったニューラル ネットワークの構築**

**概要**

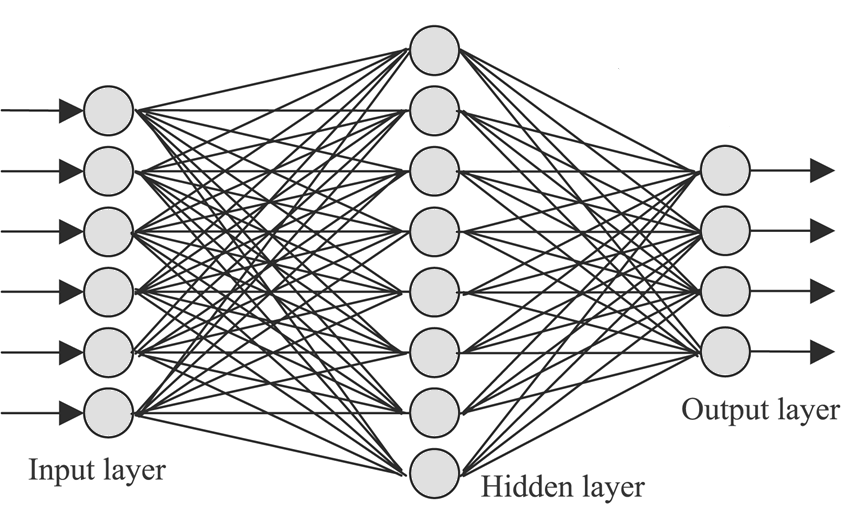
[Microsoft Cognitive Toolkit](https://www.microsoft.com/en-us/research/product/cognitive-toolkit/) (旧称: Computational Network Toolkit (CNTK)) は、ディープ ラーニング アプリケーション開発用の、強力なオープンソースの無料ツール セットです。当初、マイクロソフトのコンピューター サイエンティストが自らの研究を支援するために開発したもので、のちに、Bing やその他の製品グループで採用され、現在はマイクロソフトのお客様に使用されています。単一 CPU の 1 台のコンピューターで実行することも、Azure の [GPU オファリング](https://azure.microsoft.com/en-us/blog/azure-n-series-preview-availability/)など、複数の CPU および NVIDIA GPU を搭載した複数のコンピューター間で効率的に拡張することも可能です。このツールキットには、C++ と Python をサポートし、音声、画像、テキスト、動画を処理するための多数のライブラリとユーティリティのほか、その使用方法を示すためのサンプルも用意されています。ツールキットの開発ストーリーや使用法、機能の詳細については、次の Web サイトを参照してください。<https://blogs.microsoft.com/next/2016/10/25/microsoft-releases-beta-microsoft-cognitive-toolkit-deep-learning-advances>

Microsoft Cognitive Toolkit は、[ニューラル ネットワーク](https://en.wikipedia.org/wiki/Artificial_neural_network)による機械学習を軸としています。機械学習により、アルゴリズムでは識別が難しい、データ内のパターンをコンピューターで識別できます。猫が写っている画像を識別するアルゴリズムを記述することを想像してみてください。そのアルゴリズムでは画像をスキャンして、とがった耳、ひげ、縦長の瞳孔など、猫の特徴を探すことになるでしょう。しかし、そうした特徴を個々のピクセルやピクセルのグループを調べて識別するのは難しいはずです。一方、数百、数千もの猫の画像でトレーニングした機械学習モデルでは、データのパターンが何を表すかを明確に理解しなくても、データのパターンから猫を識別するように「学習」できるのです。機械学習は日常生活に関わり、産業界でも、クレジットカードの不正取引の検出、オンライン ショッピングのおすすめの作成など、幅広く使われています。

[](https://github.com/MSFTImagine/computerscience/blob/master/Workshop/16.Cognitive%20Toolkit/Images/cats.jpg)

*猫の特徴 (画像: ウィキメディア コモンズ)*

画像処理に使用される場合、ニューラル ネットワークは入力された画像をまず上記の図のように各範囲に分解します。これらの範囲は、ニューラル ネットワークで一連の*入力ノード*を構成します。これらのノードは、確率に基づくエッジを介して多数の*隠しノード*に接続されます。各入力範囲には、他のノードからの入力と組み合わせてエッジの確率から数学的に生成された値が割り当てられます。これらのノードからのシグナルは別のノードにフィードされ、確率的結果を指定する出力ノードに達するまで、ネットワークを横断します。

[](https://github.com/MSFTImagine/computerscience/blob/master/Workshop/16.Cognitive%20Toolkit/Images/neural-network.png)

*非表示の層が 1 つのニューラル ネットワーク*

ニューラル ネットワークでは、1 セットの入力でトレーニングし、別のセットの入力でテストします。手書き文字認識は、よくある応用事例の 1 つです。手書き文字はその人独特のものでありながら、画像で検出可能な一般的な特徴があります。

[MNIST データベース](http://yann.lecun.com/exdb/mnist/)は、手書き文字認識モデルのトレーニングと評価で一般的なデータセットです。このデータベースには、高校生から集められ、スキャンおよび正規化された 0 ～ 9 の数字の画像が、60,000 件登録されています。また、モデルの正確性を評価するための 10,000 件のテスト画像のセットも含まれています。このデータセットを使用して多数の学術論文が公開され、そのそれぞれがニューラル ネットワークを最適化して、手書きの数字を識別する際にポジティブが多く、エラーが少なくなるように追求しています。MNIST データベースを中心として構築したモデルでは、各画像を領域分割して、それをニューラル ネットワークにフィードして処理することがほとんどです。このコンテキスト、特に MNIST データでのニューラル ネットワークの使用方法の詳細説明については、<http://neuralnetworksanddeeplearning.com/chap1.html> を参照してください。

このラボでは、MNIST データベースを使用して、一対のニューラル ネットワークをトレーニングおよびテストして、トレーニングしたモデルを使用して手書きの数字の認識を行います。

**目的**

このハンズオン ラボでは、以下の方法について学習します。

* Microsoft Cognitive Toolkit をインストールする
* MNIST データセットを使用して、ニューラル ネットワークをトレーニングおよびテストする
* CNTK 構成ファイルを編集して、カスタム データセットを入力する

**前提条件**

このハンズオン ラボを完了するには、以下が必要です。

* テキスト エディターまたはプログラム エディター
* 64 ビット Linux または 64 ビット Windows がインストールされたコンピューター

64 ビット Linux または 64 ビット Windows コンピューターがない場合は、[Azure で Linux 仮想マシン](https://docs.microsoft.com/ja-jp/azure/virtual-machines/virtual-machines-linux-quick-create-portal)を作成し、VM を使用してラボを実行できます。Azure サブスクリプションがない場合は、[無料試用版にサインアップ](https://azure.microsoft.com/ja-jp/free/?WT.mc_id=AEDA13C29)します。

**演習**

このハンズオン ラボでは次の演習を行います。

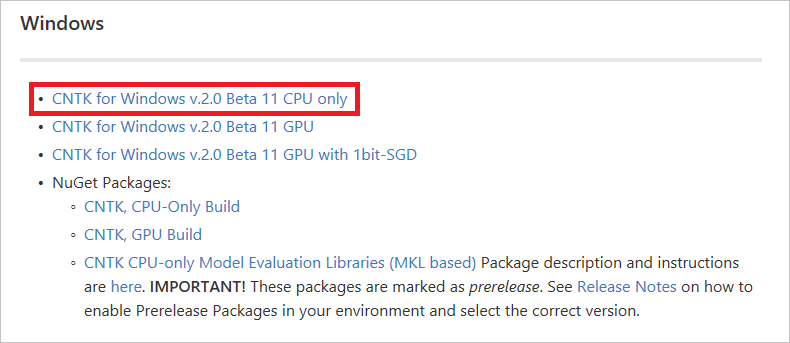
* [演習 1:](https://github.com/MSFTImagine/computerscience/blob/master/Workshop/16.Cognitive%20Toolkit/Microsoft%20Cognitive%20Toolkit%20HOL.md#Exercise1)  [Cognitive Toolkit のインストール (Windows)](https://github.com/MSFTImagine/computerscience/blob/master/Workshop/16.Cognitive%20Toolkit/Microsoft%20Cognitive%20Toolkit%20HOL.md#Exercise1)
* [演習 2:](https://github.com/MSFTImagine/computerscience/blob/master/Workshop/16.Cognitive%20Toolkit/Microsoft%20Cognitive%20Toolkit%20HOL.md#Exercise2)  [Cognitive Toolkit のインストール (Linux)](https://github.com/MSFTImagine/computerscience/blob/master/Workshop/16.Cognitive%20Toolkit/Microsoft%20Cognitive%20Toolkit%20HOL.md#Exercise2)
* [演習 3:](https://github.com/MSFTImagine/computerscience/blob/master/Workshop/16.Cognitive%20Toolkit/Microsoft%20Cognitive%20Toolkit%20HOL.md#Exercise3)  [トレーニングおよびテスト用データのインストール](https://github.com/MSFTImagine/computerscience/blob/master/Workshop/16.Cognitive%20Toolkit/Microsoft%20Cognitive%20Toolkit%20HOL.md#Exercise3)
* [演習 4:](https://github.com/MSFTImagine/computerscience/blob/master/Workshop/16.Cognitive%20Toolkit/Microsoft%20Cognitive%20Toolkit%20HOL.md#Exercise4)  [一対のニューラル ネットワークのトレーニングおよびテスト](https://github.com/MSFTImagine/computerscience/blob/master/Workshop/16.Cognitive%20Toolkit/Microsoft%20Cognitive%20Toolkit%20HOL.md#Exercise4)
* [演習 5:](https://github.com/MSFTImagine/computerscience/blob/master/Workshop/16.Cognitive%20Toolkit/Microsoft%20Cognitive%20Toolkit%20HOL.md#Exercise5)  [手書き文字のカスタム サンプルの入力](https://github.com/MSFTImagine/computerscience/blob/master/Workshop/16.Cognitive%20Toolkit/Microsoft%20Cognitive%20Toolkit%20HOL.md#Exercise5)
* [演習 6:](https://github.com/MSFTImagine/computerscience/blob/master/Workshop/16.Cognitive%20Toolkit/Microsoft%20Cognitive%20Toolkit%20HOL.md#Exercise6)  [独自の手書き文字サンプルの生成 (オプション)](https://github.com/MSFTImagine/computerscience/blob/master/Workshop/16.Cognitive%20Toolkit/Microsoft%20Cognitive%20Toolkit%20HOL.md#Exercise6)

このラボの推定所要時間: **45** 分

**演習 1: Cognitive Toolkit のインストール (Windows)**

Microsoft Cognitive Toolkit のインストールは、事前にコンパイルされたバイナリを GitHub で使用できる場合、比較的単純です。ツールキットには Windows および Linux 用のインストール スクリプトが用意されています。この演習では、Windows 用スクリプトを使用して、Windows コンピューターにツールキットをインストールします。Linux を実行している場合は、この演習をスキップして[演習 2](https://github.com/MSFTImagine/computerscience/blob/master/Workshop/16.Cognitive%20Toolkit/Microsoft%20Cognitive%20Toolkit%20HOL.md#Exercise2) に進んでください。

1. [GitHub の CNTK リリース サイト](https://github.com/Microsoft/CNTK/releases)にアクセスして、ページの [Windows] セクションでツールキットの CPU 専用バージョンの最新版をダウンロードします。使用許諾契約書に同意するよう求められたら、[はい] を選択します。

[](https://github.com/MSFTImagine/computerscience/blob/master/Workshop/16.Cognitive%20Toolkit/Images/windows-download.png)

*Windows 用 CNTK のダウンロード*

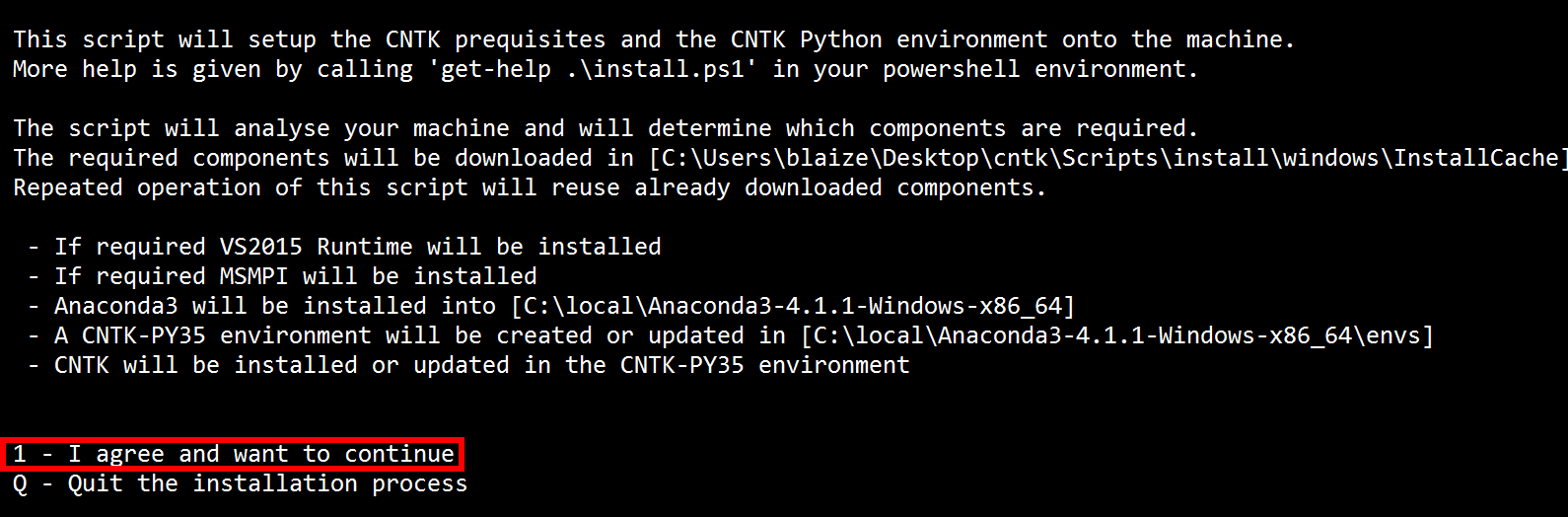
1. ダウンロードした zip ファイルを開き、「cntk」という名前のフォルダーをデスクトップまたは任意の場所にコピーします。
2. 管理者モードで [コマンド プロンプト] ウィンドウを開き,cdコマンドを使用してcntkディレクトリに切り替えて下さい。

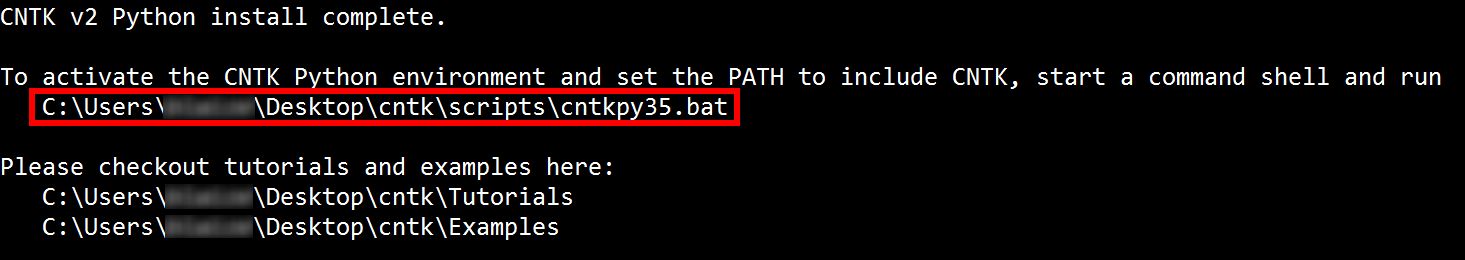
そして次のコマンドを打って下さい.

cd Scripts\install\windows

次に,以下のコマンドを使用して,CNTK及び依存ファイルをインストールして下さい

install.bat

1. プロンプトが表示され、続行するかどうかを尋ねらるたび、agree,yesといった、肯定的な解答を行って下さい。[](https://github.com/MSFTImagine/computerscience/blob/master/Workshop/16.Cognitive%20Toolkit/Images/i-agree.png)スクリプトはいくつかのライブラリをダウンロードし、インストールします。これには数分かかる場合があります。スクリプトが終了すると、CNTK Python環境を有効にする.batファイルへのパスが表示されます。（下記画像の赤で囲った部分）このコマンドをクリップボードにコピーします。



*環境を有効化するコマンドのコピー*

1. クリップボードのコマンドを、[コマンド プロンプト] ウィンドウに貼り付け、実行します。

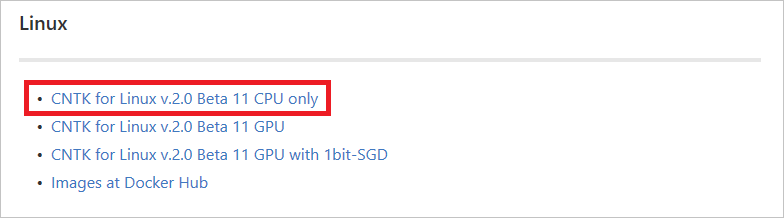
プロンプトが、CNTK Python 環境のものに変更されます。これで、CNTK がインストールされ、使用する準備ができました。このコマンドはツールキットを使用するたびに実行して CNTK Python 環境を開始する必要があります。コマンドをテキスト ファイルにコピーしておくと、あとで簡単に使用できます。

この後は、[演習 3](https://github.com/MSFTImagine/computerscience/blob/master/Workshop/16.Cognitive%20Toolkit/Microsoft%20Cognitive%20Toolkit%20HOL.md#Exercise3) にスキップします。演習 2 は Linux ユーザーのみを対象としています。

**演習 2: Cognitive Toolkit のインストール (Linux)**

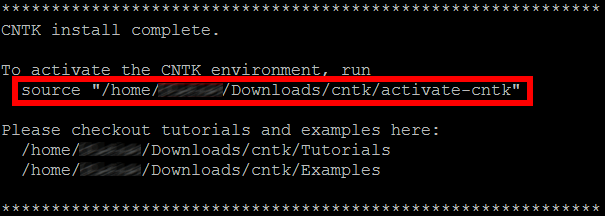
Microsoft Cognitive Toolkit のインストールは、事前にコンパイルされたバイナリを GitHub で使用できる場合、比較的単純です。ツールキットには Windows および Linux 用のインストール スクリプトが用意されています。この演習では、Linux 用スクリプトを使用して、Linux コンピューターにツールキットをインストールします。

1. [GitHub の CNTK リリース サイト](https://github.com/Microsoft/CNTK/releases)にアクセスして、ページの [Linux] セクションでツールキットの CPU 専用バージョンの最新版をダウンロードします。使用許諾契約書に同意するよう求められたら、[はい] を選択します。

[](https://github.com/MSFTImagine/computerscience/blob/master/Workshop/16.Cognitive%20Toolkit/Images/linux-download.png)

*Linux 用 CNTK のダウンロード*

1. ダウンロードの完了後、[Terminal] ウィンドウを開きます。これは通常、使用するディストリビューションの [Applications] メニューに表示されるか、**Ctrl+Alt+F1** キーを押して開始できます。
2. [Terminal] ウィンドウで「Downloads」ディレクトリに移動します。
3. cd ~/Downloads
4. 次のコマンドを使用して、**CNTK-2-X-Linux-64bit-CPU-Only.tar.gz** ファイルを解凍します。
5. tar -xvzf CNTK-2-x-Linux-64bit-CPU-Only.tar.gz
6. 次のコマンドを使用して、Linux インストール フォルダーに切り替えます。
7. cd cntk/Scripts/install/linux/
8. 次のコマンドを使用して、インストール スクリプトを開始します。sudo パスワードを求められたら、入力して **Enter** キーを押します。
9. ./install-cntk.sh
10. スクリプトにより多数のライブラリがダウンロードされ、コンピューターにインストールされます。これは、完了するまでに数分かかることがあります。スクリプトが終了したら、CNTK Python 環境を有効化するコマンドが表示されます。このコマンドをクリップボードにコピーします。

[](https://github.com/MSFTImagine/computerscience/blob/master/Workshop/16.Cognitive%20Toolkit/Images/source-command.png)

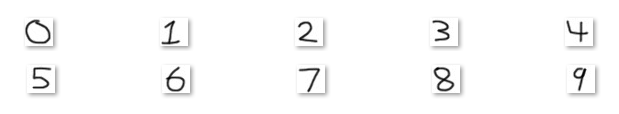
*環境を有効化するコマンドのコピー*

1. クリップボードのコマンドを、[Terminal] ウィンドウに貼り付け、実行します。

プロンプトが、CNTK Python 環境のものに変更されます。これで、CNTK がインストールされ、使用する準備ができました。このコマンドはツールキットを使用するたびに実行して CNTK Python 環境を開始する必要があります。コマンドをテキスト ファイルにコピーしておくと、あとで簡単に使用できます。

**演習 3: トレーニングおよびテスト用データのインストール**

ニューラル ネットワークのトレーニングおよびテストで比較的一般的なデータセットの 1 つに、[MNIST データベース](http://yann.lecun.com/exdb/mnist/)があります。これには、60,000 件の 0 ～ 9 の数字の手書きサンプルのトレーニング用データセットが含まれており、それぞれが 28 x 28 のグレースケール グリッドで正規化およびセンタリングされています。さらに 10,000 件の数字のテスト用サンプルが含まれています。文字は、白い背景に黒のインクを使用して書かれています。この演習では、トレーニングおよびテスト用データをダウンロードして、一連のニューラル ネットワークで使用する準備をします。

[](https://github.com/MSFTImagine/computerscience/blob/master/Workshop/16.Cognitive%20Toolkit/Images/digits.png)

*SMNIST データベースの数字サンプル*

1. 前の演習で開いた [コマンド プロンプト] または [Terminal] ウィンドウで、次のコマンドを入力して、ツールキットの「MNIST」フォルダーに移動します。

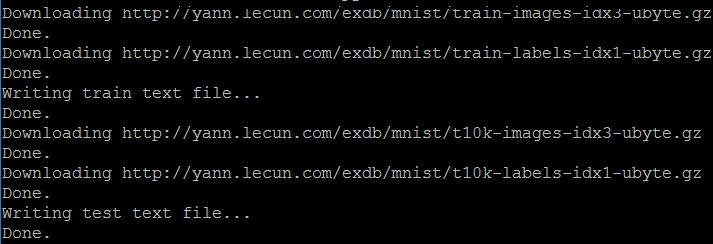
**Windows:**

cd ..\..\..\Examples\Image\DataSets\MNIST

**Linux:**

cd ../../../Examples/Image/DataSets/MNIST

1. 今度は、次のコマンドを実行して、MNIST の画像をダウンロードおよびインストールします。
2. python install\_mnist.py
3. スクリプトが終了するのを待ちます。ダウンロード後、データセットは CNTK で使用できるように、バイナリ形式からテキスト形式に変換されます。入力に使用されるこれらのテキスト ファイルの詳細については、[演習 5](https://github.com/MSFTImagine/computerscience/blob/master/Workshop/16.Cognitive%20Toolkit/Microsoft%20Cognitive%20Toolkit%20HOL.md#Exercise5) を参照してください。

[](https://github.com/MSFTImagine/computerscience/blob/master/Workshop/16.Cognitive%20Toolkit/Images/install-dataset.png)

*インストール スクリプトの完了*

スクリプトにより、テキスト ファイルが 2 つ作成されます。1 つはトレーニング用、もう 1 つはテスト用です。トレーニング データはテキスト ファイルで、ファイル名は **Train-28x28\_cntk\_text.txt** です。テストデータは **Test-28x28\_cntk\_text.txt** です。次に進む前に、現在いるディレクトリに両方のファイルがあることを確認します。

**演習 4: 一対のニューラル ネットワークのトレーニングおよびテスト**

Microsoft Cognitive Toolkit には、構成ファイルがいくつか用意されており、1 つのコマンドでさまざまな種類のニューラル ネットワークを作成、トレーニング、テストできるようになっています。最も単純なネットワーク タイプの 1 つが「非表示層が 1 つ」のネットワークで、ディープ ニューラル ネットワークではなく、隠しノードの層が 1 つだけ含まれているネットワークとなっています。CNTK が標準でサポートするもう 1 つのニューラル ネットワーク タイプは、“コンボリューション層が 1 つ”の[コンボリューショナル ニューラル ネットワーク](https://en.wikipedia.org/wiki/Convolutional_neural_network)です。コンボリューショナル ネットワークは、画像認識でよく使用されます。

この演習では、MNIST 画像データを使用して非表示層が 1 つのネットワークとコンボリューションが 1 つのネットワークをトレーニングおよびテストし、結果を比較します。

1. 前の演習で開いた [コマンド プロンプト] または [Terminal] ウィンドウで、次のコマンドを入力して、ツールキットの「GettingStarted」フォルダーに移動します。

**Windows:**

cd ..\..\GettingStarted

**Linux:**

cd ../../GettingStarted

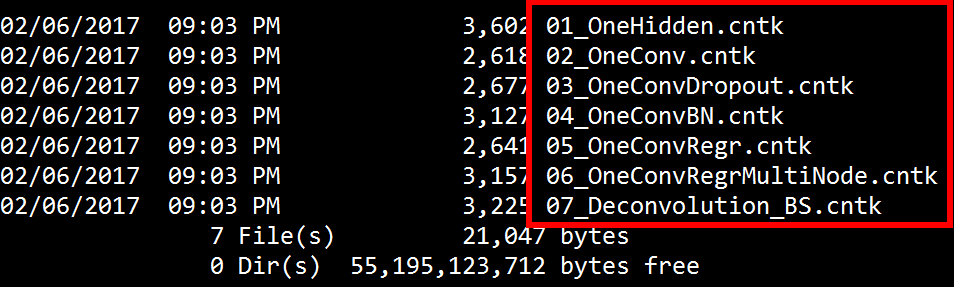
1. 「GettingStarted」フォルダーには複数の .cntk ファイルが含まれ、それぞれが特定の種類のネットワークに対応し、それぞれが MNIST データセットを対象としています。各 .cntk ファイルには、ニューラル ネットワークの作成、MNIST データでのトレーニング、テストについての説明が含まれています。次のコマンドを使用して、ディレクトリの .cntk ファイルを一覧表示します。

**Windows:**

dir \*.cntk

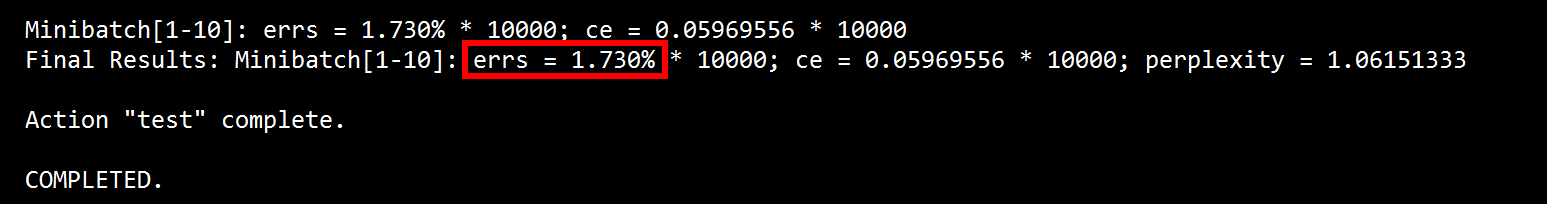
**Linux:**

ls -l \*.cntk

[](https://github.com/MSFTImagine/computerscience/blob/master/Workshop/16.Cognitive%20Toolkit/Images/cntk-files.png)

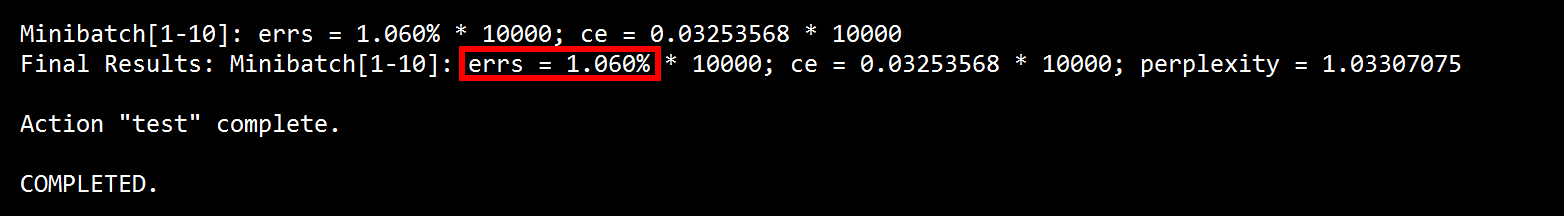
*ニューラル ネットワークのテストおよびトレーニング用 .cntk ファイル*

1. 次のコマンドを使用して非表示層が 1 つのネットワークをトレーニング、テストします。
2. cntk configFile=01\_OneHidden.cntk
3. トレーニングとテストが完了するまで待ちます。トレーニングとテストの処理による出力が画面に表示されます。最後にテストのまとめが表示され、テスト中に発生したエラーの割合を示す **エラー**値などが示されます。割合が低いということは、ニューラル ネットワークでより多くの数字をテスト データセット内で正しく識別できたということを意味します。

[](https://github.com/MSFTImagine/computerscience/blob/master/Workshop/16.Cognitive%20Toolkit/Images/one-hidden.png)

*非表示層が 1 つのニューラル ネットワークのテスト*

1. 今度は、次のコマンドを使用して、コンボリューションが 1 つのネットワークをトレーニングおよびテストします。
2. cntk configFile=02\_OneConv.cntk
3. トレーニングとテストが完了するまで待ちます。このネットワークのエラー率が低いことに注意してください。これは、最初のネットワークよりも多く、テスト用の数字を正しく識別したことを意味します。

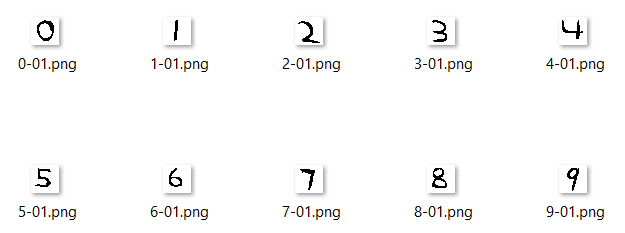
[](https://github.com/MSFTImagine/computerscience/blob/master/Workshop/16.Cognitive%20Toolkit/Images/one-conv.png)

*コンボリューションが 1 つの ニューラル ネットワークのテスト*

時間が許す限り、ディレクトリ内の他の .cntk ファイルのいくつかを使用して、別のニューラル ネットワークをトレーニングおよびテストして、非表示層が 1 つのネットワークや、コンボリューションが 1 つのネットワークと結果を比較してみてください。

**演習 5: 手書き文字のカスタム サンプルの入力**

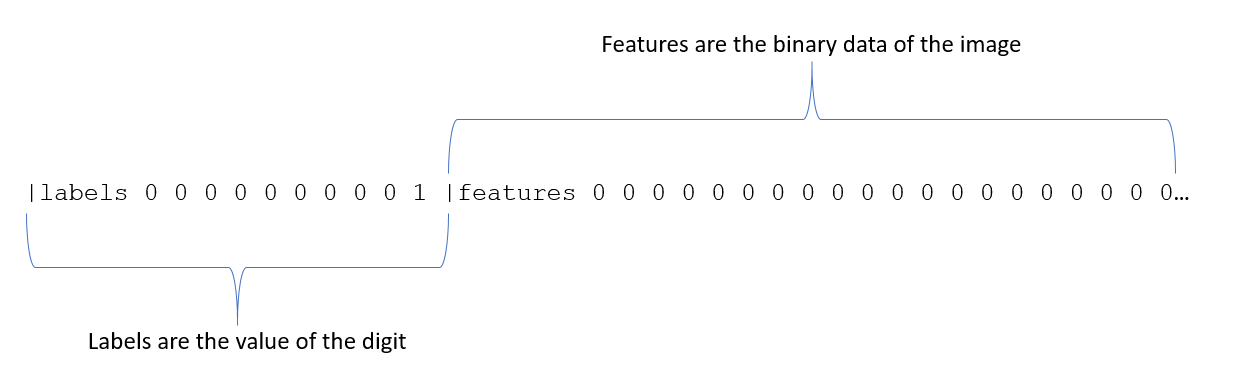
このラボの「resources」フォルダーには、手書き文字認識のテストに使用できるカスタム画像のセットや、CNTK でのテストに必要なテキスト ファイルを作成する Python スクリプトが入っています。これらの画像は MNIST データセットの一部ではありませんが、この演習用に生成されたものです。

[](https://github.com/MSFTImagine/computerscience/blob/master/Workshop/16.Cognitive%20Toolkit/Images/test-images.png)

*ニューラル ネットワークのテスト用カスタム画像*

この演習では、これらの画像を使用して、前の演習でトレーニングしたネットワークで、テスト用データセットの一部ではない手書きの数字を認識する精度を確かめます。

1. [コマンド プロンプト] または [Terminal] ウィンドウで、このラボの「resources」ディレクトリに移動します。そして、次のコマンドを使用して、**images2cntk.py** という名前のスクリプトを実行します。これにより、**Custom-Test-28x28\_cntk\_text.txt** という名前のファイルが作成されます。
2. python images2cntk.py
3. 任意のテキスト エディターまたはプログラム エディターで **Custom-Test-28x28\_cntk\_text.txt** を開き、内容を確認します。このファイルには 10 行あり、各行にテスト画像が 1 つ含まれています。各行には「labels」という名前のセクションと、「features」という名前のセクションがあります。「labels」のデータは、0 ～ 9 の数字の実際の値を示しています。最初の値は 0 に対する 1 または 0、2 番目は 1 に対する 1 または 0、というようになっています。数字の 7 の場合、「labels」のすべての値は、8 番目を除いて 0 となり、8 番目は 1 となります。「features」のデータは画像そのものを表します。28 x 28 の配列の各ピクセルに 1 つで 784 の値があり、それぞれの値は 0 ～ 255 のグレースケール値で、0 は空白、255 は塗りつぶし (「ink」) を表します。

[](https://github.com/MSFTImagine/computerscience/blob/master/Workshop/16.Cognitive%20Toolkit/Images/sample-data.png)

*サンプル データを含むテキスト ファイル*

1. **cd** コマンドを使用して「GettingStarted」ディレクトリに戻ります。前の演習で **cntk** コマンドを実行したディレクトリです。
2. **01\_OneHidden.cntk** のコピーを作成し、名前を **01\_OneHidden\_Custom.cntk** にします。次に、**01\_OneHidden\_Custom.cntk** をテキスト エディターまたはプログラム エディターで開きます。
3. ファイルの末尾までスクロールし、「# TEST CONFIG」というコメントを探します。このコメントの下にある「reader」セクションは、CNTK でテスト データを探す場所を示します。このセクションは、ツールキットでインストールされた時点では、**Test-28x28\_cntk\_text.txt** ファイルを指定しています。**file =** 行を編集して、この演習のステップ 1 で作成した **Custom-Test-28x28\_cntk\_text.txt** ファイルのフルパスに変更します。次に、Windows のパス名を使用した例を示します。

reader = {

readerType = "CNTKTextFormatReader"

file = "C:\Labs\CNTK\resources\Custom-Test-28x28\_cntk\_text.txt"

input = {

features = { dim = 784 ; format = "dense" }

labels = { dim = 10 ; format = "dense" }

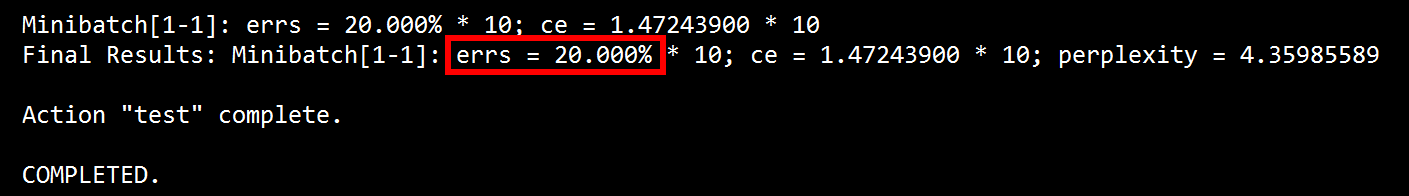
}

}

1. reader = {
2. readerType = "CNTKTextFormatReader"
3. file = "C:\Labs\CNTK\resources\Custom-Test-28x28\_cntk\_text.txt"
4. input = {
5. features = { dim = 784 ; format = "dense" }
6. labels = { dim = 10 ; format = "dense" }
7. }
8. }
9. 変更した .cntk ファイルを保存します。
10. 変更した .cntk ファイルで **cntk** コマンドを実行します。トレーニング済みのモデルがディスクに保存されているので、テスト スクリプトがより高速に実行されるようになります。CNTK では、再生成を行わず、トレーニング済みのモデルを使用します。

cntk configFile=01\_OneHidden\_Custom.cntk

1. cntk configFile=01\_OneHidden\_Custom.cntk
2. カスタム画像のエラー率に注意してください。

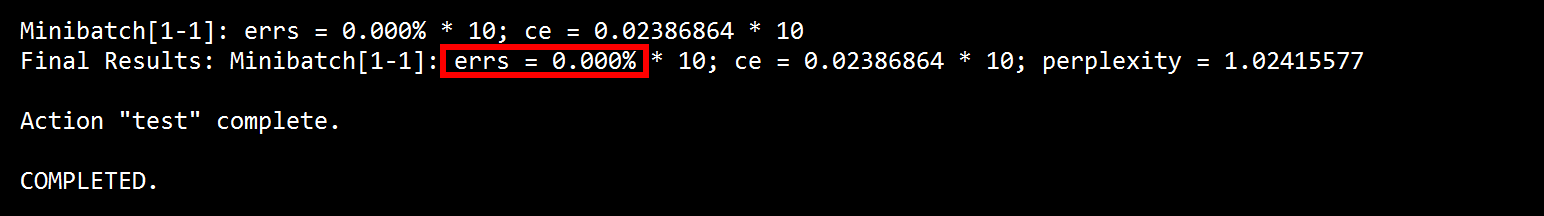
[](https://github.com/MSFTImagine/computerscience/blob/master/Workshop/16.Cognitive%20Toolkit/Images/one-hidden-custom.png)

*カスタム画像での非表示層が 1 つのネットワークのテスト*

1. ここで、**02\_OneConv.cntk** のコピーを作成し、**02\_OneConv\_Custom.cntk** という名前を付けて、テスト画像を使用するように編集します。それから次のコマンドを使用して、テスト画像でコンボリューションが 1 つのネットワークを実行します。

cntk configFile=02\_OneConv\_Custom.cntk

1. cntk configFile=02\_OneConv\_Custom.cntk
2. カスタム画像のエラー率に注意してください。テスト画像の数字の認識率が高いのは、どちらのネットワークでしょうか。

[](https://github.com/MSFTImagine/computerscience/blob/master/Workshop/16.Cognitive%20Toolkit/Images/one-conv-custom.png)

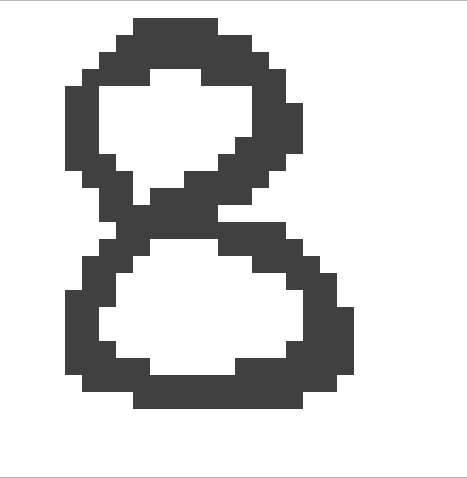
*カスタム画像でのコンボリューションが 1 つのネットワークのテスト*

これらのテストは、同じ入力データでトレーニングした場合でも、ニューラル ネットワークが異なれば、生成される結果も異なる可能性があることを示しています。ここでは、テスト画像の数字の認識については、コンボリューションが 1 つのネットワークの方が優れていました。

**演習 6: 独自の手書き文字サンプルの生成 (オプション)**

希望する場合は、任意のビットマップ エディターを使用して、独自のテスト画像を生成し、作成したニューラル ネットワークで実行することができます。ここでは、その方法について説明します。

1. 背景が白の 28 x 28 ビットマップを作成します。
2. ズームインして、数字を描きます。

[](https://github.com/MSFTImagine/computerscience/blob/master/Workshop/16.Cognitive%20Toolkit/Images/edit-custom-image-zoomed.png)

*8 の描画*

1. 画像を、このラボの「resources」ディレクトリの「input-images」フォルダーに保存します。ファイル名は **X-YY.png** にします。ここで、X は数字の値、YY は 01 以外の任意の数字にします (テスト画像で使用するサフィックスはこのラボで提示されます)。たとえば、上記の画像の名前は **8-02.png** となります。
2. ステップ 1 ～ 3 を繰り返して、テスト画像をさらに作成します。
3. 最後に、[演習 5](https://github.com/MSFTImagine/computerscience/blob/master/Workshop/16.Cognitive%20Toolkit/Microsoft%20Cognitive%20Toolkit%20HOL.md#Exercise5) の手順を繰り返して、画像データを含むテキスト ファイルを再生成して、新しい画像セットでテストを再実行します。

ニューラル ネットワークは楽しいですね! Microsoft Cognitive Toolkit があれば、作成も簡単です。

**まとめ**

ニューラル ネットワークは、さまざまな種類のメディアにパターン認識を適用するのに役立ちます。このラボでは、Microsoft Cognitive Toolkit (旧称: CNTK) を使用して、ニューラル ネットワークをトレーニングし、MNIST データベースを使用して手書き文字認識を行いました。ツールキットには、音声認識、画像分類、言語理解などのライブラリとサンプル、さらに、ニューラル ネットワークと組み合わせたアプリケーションの構築に使用できる API 一式も含まれています。詳細および、その他の学習サンプルについては、<https://www.microsoft.com/en-us/research/product/cognitive-toolkit/model-gallery/> を参照してください。

機械学習に不慣れな場合、その仕組みや使用されるタスクの種類などの詳細については、<https://medium.com/@ageitgey/machine-learning-is-fun-80ea3ec3c471> の記事を参照してください。特に、[第 2 部](https://medium.com/@ageitgey/machine-learning-is-fun-part-2-a26a10b68df3)および[第 3 部](https://medium.com/@ageitgey/machine-learning-is-fun-part-3-deep-learning-and-convolutional-neural-networks-f40359318721)では、ニューラル ネットワークについて取り上げており、ニューラル ネットワークとは何か、その仕組みについて理解するのに非常に役立ちます。

Copyright 2016 Microsoft Corporation. All rights reserved. 特に注記がない限り、これらの資料は MIT ライセンスの使用条件下でライセンス供与されます。これらの資料は、MIT ライセンスに従い、プロジェクトに最も適切と判断された場合は使用することができます。このライセンスの使用条件については、<https://opensource.org/licenses/MIT> を参照してください。