

Final

by Yifei Sun

Submission date: 30-Mar-2023 09:29AM (UTC+0100)

Submission ID: 200975544

File name: 39006_Yifei_Sun_Final_495066_562903098.pdf (508.39K)

Word count: 3405

Character count: 18021

Classification using ML methods and Neural Networks

Yifei Sun

March 2023

1 Abstract

This study presents a comparison between a neural network and a random forest classifier for image classification tasks using the fashion-mnist and cifar-10 datasets. The hyperparameters of the models were optimized based on a literature review, and the performance of the models was evaluated using various metrics, including accuracy, F1 score, precision and confusion matrix. The experimental results show that the neural network classifier outperforms the random forest classifier in each evaluation index in these two classification tasks. Furthermore, the confusion matrix analysis reveals that the neural network model achieves better classification performance across all classes. Overall, This study proves that neural networks are more effective than random forests in image classification tasks, and provides insights on how to select appropriate machine learning models for similar classification tasks.

2 Introduction

2.1 Neural Network Introduction

The concept of neural networks was proposed inspired by the human neural structure. Fukushima, K[1] was the first person to apply animal neural networks to computers. He proposed a neural network structure that

includes convolutional layers and pooling layers, which laid the foundation for subsequent artificial intelligence research. Hecht-Nielsen[2] proposed the backpropagation function, which is typically used to train neural networks. This algorithm updates model parameters, including the weight and bias values of each neuron, based on the error between the model output and the true label. Each layer of neurons in a neural network converts input signals to output signals and passes them on to the next layer. These neurons use non-linear activation functions to achieve this transformation. Some important functions and formulas in neural networks are:

Neuron output formula (Sigmoid): $f(x) = \frac{1}{1+e^{-x}}$ where x is the weighted sum of neuron inputs.

Neuron output formula (ReLU): $f(x) = \max(0, x)$ where x is the weighted sum of neuron inputs.

Neuron output formula (tanh): $f(x) = \tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$ where x is the weighted sum of neuron inputs.

Softmax function: $\text{softmax}(x_i) = \frac{e^{x_i}}{\sum_{j=1}^n e^{x_j}}$ The softmax function maps the input vector $\mathbf{x}$ to a probability distribution.

Loss Function: The loss function is used to measure the difference between the model output and the actual label. It includes mean square error, root mean square error, and others. The cross entropy loss function used in this article is:

Cross-entropy loss function: $L = -\sum_{i=1}^n y_i \log \hat{y}_i$, where y is the true label and $\hat{y}$ is the predicted label.

2.2 Random Forest Introduction

The random forest model was proposed by Leo Breiman [3] in 2001, and is an integrated learning method that combines multiple decision trees to improve prediction accuracy and generalization ability. Randomness is introduced in the random forest model, including random sampling of samples and random selection of features, to generate multiple decision trees under different randomness and combine them to obtain the final prediction result. In constructing decision trees, random forest randomly selects a certain number of samples and features for training each time from the training set,

so that the training sets of each decision tree are not completely the same, thereby reducing the risk of overfitting. At the same time, the decision tree splitting process also increases the randomness of the model, making the decision trees more diverse, further improving the model's robustness and generalization ability.

The definitions of two hyperparameters used in this article are as follows: `n_estimators` is the number of trees; `random_state` is a random seed used to control the randomness in each run of the random forest. If the same seed is specified, the results will be consistent across runs. If not specified, the results will vary across runs.

3 Related work in the literature

Martin Abadi et al.[4] described the TensorFlow system and some TensorFlow-based models, and demonstrated how models compiled with TensorFlow can be used for high-performance training. TensorFlow has made rapid progress in machine learning and is widely used by researchers due to its large user base and many pre-existing models. In this study, I will use two classifiers, namely, random forest and neural network, to classify the Fashion-Mnist and Cifar10 image datasets, respectively, and their accuracies were evaluated. Samia et al.[5] used deep learning and machine learning methods to classify the Fashion mnist dataset, and evaluated the performance of algorithms using machine learning algorithms such as Support Vector Machine, random forests, KNN, and deep learning models such as CNN. They used accuracy and confusion matrix as evaluation metrics and obtained good results. Similarly, in this study, these two metrics were used to evaluate the performance of the two models. Ertam and Aydın[6] conducted a classification task on the mnist dataset using TensorFlow and tested the accuracy of the classifier by selecting different activation functions on the system, including ReLu, tanH, and sigmoid. They used SoftMax as the classifier function, and their conclusion was that the ReLu function had the highest accuracy. In this study, these three activation functions were also compared to ensure that the neural network model had the highest

accuracy. Huang et al.[7] used convolutional neural networks via TensorFlow to recognize handwritten digits in the MNIST dataset. They varied hyperparameters such as dropout rate, epochs, and different numbers of neurons in the experiment, with the highest detection accuracy of 0.2 and 50 periods, respectively. It is worth noting that the dataset used in this paper is more complex than the MNIST dataset, so more epochs may be needed. Zeng et al.[8] designed a gesture recognition framework based on TensorFlow. Although the convolutional neural network used in their study differs from the research topic of this paper, in one of their chapters, they emphasized that using GPU mode can speed up the running time. Similarly, this study was also conducted in GPU mode and found that the running efficiency was greatly improved compared to before. For the literature review on random forest models, I searched for a paper on the application of random forest in classification models, and found Shaik, A.B.'s [9] survey, which compared the performance of various random forest models and introduced the advantages of random forest classifiers, including their fast construction and prediction speed, not requiring any cross-validation, and being fully parallelizable. This article will focus on the part related to neural network models, as the random forest model uses a relatively simple structure, which will not be further elaborated.

4 Methodology

4.1 Establish neural network model

This study uses the anaconda3 environment and a non-convolutional neural network model based on TensorFlow. The specific steps are as follows, and the code parts are shown in Fig1 and Fig2.

1. Import necessary modules and libraries, including TensorFlow, NumPy, Scikit-learn, etc.
2. Load the Fashion-MNIST dataset, which includes a training subset and a test subset, and all images are a 28x28 pixel grayscale images. Define category names, namely the names of the 10 categories in the Fashion-MNIST dataset.

3. Split the imported Fashion Mnist training set into training sets and validation sets, and normalize the pixel values to values between 0 and 1. Normalized preprocessing of data helps neural network models better learn features while improving model performance. Then, perform one-hot encoding on the target labels for use during model training.

4. Define a NN model with two fully connected layers. The first layer is a Flatten layer that flattens the image into a one-dimensional vector. The second layer is a fully connected layer with 256 neurons, using the sigmoid activation function. The third layer is a fully connected layer with 20 neurons, using the softmax activation function.

5. Compile the model, choose the Adam optimizer, use sparse categorical cross-entropy loss function, and accuracy metric.

6. Using the NN model to train the training dataset, with 30 iterations in total. In each iteration, use batch gradient descent algorithm to update the model parameters.

7. Evaluate the performance of the model on the test set, output the loss and accuracy on the test set. Predict on the test set and calculate evaluation metrics.

```

from __future__ import absolute_import, division, print_function, unicode_literals
from tensorflow import keras
import numpy as np
from sklearn.preprocessing import OneHotEncoder
from sklearn.model_selection import train_test_split
from sklearn.metrics import precision_recall_fscore_support, accuracy_score
from sklearn.metrics import confusion_matrix

# Load the Fashion-MNIST dataset
fashion_mnist = keras.datasets.fashion_mnist
(train_images, train_labels), (test_images, test_labels) = fashion_mnist.load_data()

# Define the class names
class_names = ['T-shirt/top', 'Trouser', 'Pullover', 'Dress', 'Coat', 'Sandal', 'Shirt', 'Sneaker', 'Bag', 'Ankle boot']

# Split the training set into training and validation sets
train_images, val_images, train_labels, val_labels = train_test_split(train_images, train_labels, test_size=0.2, random_state=42)

# Normalize pixel values to be between 0 and 1
train_images = train_images / 255.0
val_images = val_images / 255.0
test_images = test_images / 255.0

# One-hot encode the target labels
encoder = OneHotEncoder()
train_labels = np.array(train_labels)
val_labels = np.array(val_labels)
test_labels = np.array(test_labels)

# Define the neural network architecture
model = keras.Sequential([
    keras.layers.Flatten(input_shape=(28, 28)),
    keras.layers.Dense(128, activation='sigmoid'),
    keras.layers.Dense(10, activation='softmax')
])

# Compile the model with Adam optimizer, sparse categorical cross-entropy loss, and accuracy metrics
model.compile(optimizer='adam',
              loss='sparse_categorical_crossentropy',
              metrics=['accuracy'])

# Train the model for 10 epochs
model.fit(train_images, train_labels, epochs=10)

```

Fig 1: Neural network model

```

# Evaluate the trained model on the test set
test_loss, test_acc = model.evaluate(test_images, test_labels, verbose=2)

print("Test loss:", test_loss, "Test acc:", test_acc)

# Make predictions on the test set
y_pred = np.argmax(model.predict(test_images), axis=1)

# Compute the accuracy, precision, recall, and F1 score for the predictions
acc = accuracy_score(test_labels, y_pred)
precision, recall, f1_score, support = precision_recall_fscore_support(test_labels, y_pred, average='macro')
cm = confusion_matrix(test_labels, y_pred)

# Print the evaluation metrics
print("Accuracy:", acc)
print("Precision:", precision)
print("Recall:", recall)
print("F1 Score:", f1_score)
# Print the confusion matrix
print("Confusion Matrix:\n", cm)

```

Fig 2: Neural network model

4.2 Establish random forest model

This study utilized the anaconda3 environment and implemented a random forest model based on TensorFlow. The specific steps are as follows, with corresponding code shown in Fig3.

1. Import necessary libraries, including TensorFlow, the random forest classifier from sklearn, functions for evaluating Common evaluation indicators for machine learning, and a function for computing the confusion matrix.
2. Load the fashion MNIST dataset and normalize the training and test images to the range [0, 1].
3. Train a classification model using the random forest classifier from sklearn, using the image features and labels from the training dataset as inputs.
4. Make predictions on the test images and obtain the predicted labels, y_pred.
5. Evaluate the performance of a model on the Fashion-mnist and Cifar-10 datasets by calculating various machine learning evaluation metrics including precision, and then output the results.

```
import tensorflow as tf
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import precision_recall_fscore_support, accuracy_score
from sklearn.metrics import confusion_matrix

fashion_mnist = tf.keras.datasets.fashion_mnist
(train_images, train_labels), (test_images, test_labels) = fashion_mnist.load_data()

train_images = train_images / 255.0
test_images = test_images / 255.0

rf = RandomForestClassifier(n_estimators=500, random_state=42)
rf.fit(train_images.reshape(-1, 784), train_labels)

y_pred = rf.predict(test_images.reshape(-1, 784))

acc = accuracy_score(test_labels, y_pred)
precision, recall, f1_score, support = precision_recall_fscore_support(test_labels, y_pred, average='macro')
cm = confusion_matrix(test_labels, y_pred)
print("Accuracy:", acc)
print("Precision:", precision)
print("Recall:", recall)
print("F1 Score:", f1_score)
# Print the confusion matrix
print("Confusion Matrix:\n", cm)
```

Fig 3:

4.3 Modify the parameters and improve the model

Through learning relevant literature and the course content, there are many hyperparameters that can be modified to improve the accuracy of a NN model. Through this study, the following modifications were made to

the model. It is worth noting that some attempts actually reduced accuracy and further research and adjustments are needed.

1. Increase the number of training epochs: Initially, the model was trained for 10 epochs, but the accuracy was low. After increasing the number of epochs to 30, there was a significant improvement in accuracy, but the training time also increased significantly. The code was modified from "epochs=10" to "epochs=30".

2. Adjust the learning rate: The learning rate is a hyperparameter in the Adam optimizer that can be adjusted in the `model.compile()` function. It is worth noting that Adam has a default learning rate, and it can dynamically adjust the learning rate. A smaller learning rate can make the model converge more stably, but it may require more training time. A larger learning rate can speed up the training process, but it may cause the model to fluctuate near the optimal point. Adam optimizer was used in this study, so no adjustments were made.

3. Change the number of neurons: Adding or removing fully connected layers in each layer can affect the model's complexity and capacity. Increasing the number of neurons can improve the model's representation ability, but it may also lead to overfitting. Decreasing the number of neurons can reduce the risk of overfitting, but it may also lead to underfitting. In this study, the hidden layer parameter is set to 256 and the output layer parameter is set to 20.

4. Use different activation functions: Changing the activation function in the model can affect the model's ability to extract nonlinear features. ReLU is a preferred activation function based on previous literature reviews, but other activation functions, such as tanh or sigmoid, can also be tried. The code was modified from "activation='relu'" to "activation='tanh'" and "activation='sigmoid'".

5. Add regularization: Adding regularization to the model, such as L1 or L2 regularization, can help control the model's complexity and reduce the risk of overfitting. In this study, regularization was first added by using the kernel regularizer parameter to add regularization to each layer. L2 regularization was used, and the regularization strengths were set to 0.01

and 0.05 for comparison. The accuracy was slightly reduced compared to before, and the training time was increased. One reason is that the L2 parameter selection was not accurate enough, and the other reason is that the type of regularization was not used properly.

For the random forest model, the main adjustments were made to the number of trees and the randomness of the model. In this study, the parameters were set to n_estimators=500 and random_state=42 to ensure that the model could run properly.

4.4 Evaluation Methods

Accuracy refers to the proportion of the correct prediction of the model results to the overall prediction, and its formula is as follows:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

Precision refers to the proportion of true positive samples among those predicted as positive by the model:

$$Precision = \frac{TP}{TP + FP}$$

Recall refers to the proportion of true positive samples among all positive samples in the dataset:

$$Recall = \frac{TP}{TP + FN}$$

The F1 score is a weighted harmonic average of accuracy and recall. Since precision and recall are often in a trade-off situation, F1 score can be considered as the balance point between the two.

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

Here, TP represents true positive, which is the number of samples correctly classified as positive by the model; TN represents true negative, which is the number of samples correctly classified as negative by the model; FP represents a false positive, which is the number of samples that are incorrectly classified as positive by the model; FN represents false negative, which

is the number of samples incorrectly classified as negative by the model. The range of these values is from 0 to 1. If the value is close to 0, it indicates that the model's performance is poor. If it is close to 1, it indicates that the model's performance is good.

Confusion Matrix is a two-dimensional table used to evaluate the performance of a classification algorithm, typically in supervised learning. It presents the classification results of samples in four categories: true positives (TP), false positives (FP), true negatives (TN), and false negatives (FN), with the actual class as rows and the predicted class as columns.

5 Comparison of models using datasets

5.1 Classification of Fashion MNIST Dataset

Fashion MNIST is a clothing image classification dataset based on the MNIST dataset, used for machine learning and computer vision research. It consists of 70,000 28x28 grayscale images in 10 clothing categories, including T-shirts/tops, trousers, pullovers, dresses, coats, sandals, shirts, sneakers, bags, and ankle boots. As previously mentioned, a NN model based on TensorFlow was used for classifying the Fashion-MNIST dataset, and after 30 training iterations, the accuracy was as follows. Compared to the random forest model, the neural network model had more adjustable hyperparameters, resulting in higher accuracy. Fig 4 and Fig 5 show the accuracy, precision, F1 score, and confusion matrix for the NN model and the RF model, respectively.

```

Accuracy: 0.8882
Precision: 0.8912345408667125
Recall: 0.8882
F1 Score: 0.8887307016143072
Confusion Matrix:]
[[822   3   8  25   1   2 129   0  10   0]
 [  3 979   0  10   4   0   3   0   1   0]
 [ 16   1 746  10 107   1 118   0   1   0]
 [ 15   6  10 915  16   0  32   0   6   0]
 [   1   0  55  39 795   0 103   0   7   0]
 [   2   0   0   1   0 963   0  21   2  11]
 [  91   0  49  27  48   0 772   0  13   0]
 [   0   0   0   0   0  11   0 978   0  11]
 [   3   0   4   5   3   2   0   3 979   1]
 [   0   0   0   0   0   9   1  57   0 933]]

```

Fig 4:

```

Accuracy: 0.878
Precision: 0.8768743670867233
Recall: 0.8779999999999999
F1 Score: 0.8766203048649119
Confusion Matrix:
[[862   0  11  28   3   1  84   0  11   0]
 [   3 962   3  23   2   0   5   0   2   0]
 [  12   0 803   9 112   0  59   0   5   0]
 [  17   2   9 908  30   0  32   0   2   0]
 [   0   1  89  32 826   0  50   0   2   0]
 [   0   0   0   1   0 959   0  29   1  10]
 [149   1 122  29  89   0 591   0  19   0]
 [   0   0   0   0   0  13   0 952   0  35]
 [   0   2   5   2   5   2   6   5 973   0]
 [   0   0   0   0   0   9   1  44   2 944]]

```

Fig 5:

5.2 Classification of CIFAR-10 Image Dataset

Compared to Fashion-MNIST, the CIFAR-10 image dataset is more complex. CIFAR-10 is an image classification dataset commonly used to test various machine learning models consisting of 10 different classes, each containing 6,000 32x32 color images from different angles and scales. The dataset includes 50000 training images and 10000 test images. The categories in the CIFAR-10 dataset include real-world vehicles, animals, and the images come from the network. Compared to the first task, this dataset is

highly complex. Each image in each class was collected from the internet and labeled by humans with its correct class. In this study, the accuracy of the neural network model for CIFAR-10 was 0.7, while the random forest model had a lower accuracy of only 0.48. Fig 6 and Fig 7 show the comparison of the results for the two models.

```
Accuracy: 0.7157
Precision: 0.7125993938078887
Recall: 0.7157000000000001
F1 Score: 0.7106841941246678
Confusion Matrix:]
[[791 21 39 18 14 9 11 17 45 35]
 [ 15 882 7 3 3 4 12 8 23 43]
 [ 81 9 548 52 79 72 89 42 16 12]
 [ 24 22 71 419 84 181 121 51 15 12]
 [ 23 4 64 41 653 42 71 87 12 3]
 [ 17 13 36 116 52 637 50 65 9 5]
 [ 4 7 30 37 47 25 839 5 3 3]
 [ 11 3 32 19 42 53 10 817 1 12]
 [ 71 45 9 10 5 8 8 9 820 15]
 [ 50 120 5 11 8 3 15 17 20 751]]
```

Fig 6:

```
Accuracy: 0.4872
Precision: 0.4825702383304963
Recall: 0.48719999999999997
F1 Score: 0.4826319714782884
Confusion Matrix:
[[557 32 56 17 28 16 25 26 177 66]
 [ 29 574 14 32 14 32 37 35 54 179]
 [ 97 43 334 82 135 65 128 55 28 33]
 [ 53 44 62 295 78 173 142 56 24 73]
 [ 51 16 134 49 401 52 158 90 24 25]
 [ 27 27 79 160 74 416 76 82 28 31]
 [ 8 32 68 61 98 48 613 23 8 41]
 [ 39 40 50 47 89 76 46 483 24 106]
 [ 85 82 18 22 17 39 12 22 624 79]
 [ 43 162 14 28 18 19 26 40 75 575]]
```

Fig 7:

6 Results and discussion

In the first task, we employed neural network and random forest models to classify the Fashion-mnist dataset. Results showed that the neural network model achieved an accuracy of 0.8882, slightly higher than the random forest model's accuracy of 0.878. The neural network model also exhibited a higher Precision of 0.8912, compared to the RF model Precision of 0.8769, and a higher Recall of 0.8882, compared to the RF model Recall of 0.878. The F1 Score for the NN model is 0.8887, slightly higher than the RF model F1 Score of 0.8766.

Examining the confusion matrices of the two models revealed that they both accurately classified most of the categories, but their performance deteriorated when dealing with more complex images. The neural network model achieved the lowest accuracy in classifying category 2 (Pullover), with a score of 0.746, while the random forest model achieved the lowest accuracy in classifying category 6 (Shirt), with a score of 0.591. This implies that the neural network model was more prone to misclassifying Pullovers, while the random forest model was more likely to misclassify Shirts. Moreover, both models misclassified a substantial number of category 6 (Shirt) samples as category 0 (T-shirt/top), with 91 and 149 instances, respectively.

In the second task, these two models were used for classifying the CIFAR-10 dataset, which is more complex than the previous dataset, resulting in less satisfactory performance. According to the data, the NN model outperformed the RF model in classifying the CIFAR-10 dataset. The neural network model achieved an accuracy of 0.7157, while the random forest model achieved an accuracy of 0.4872. In the confusion matrix of the NN model, the classification performance is better than that of the RF model for many categories. For instance, the classification of category 0 (airplane), the precision of the NN model is 0.79, while the precision of the RF model is 0.557. In the classification of category 6 (frog), the recall of the NN model is 0.839, while the recall of the RF model is 0.613. Therefore, based on these indicators, it can be concluded that in the classification of the CIFAR-10 dataset, neural network models perform better in the face of more complex image classification problems, while the accuracy of random

forest models is not accurate enough in the face of complex problems, possibly due to the lack of further optimization of the model and processing of the dataset.

In conclusion, although both models performed well in classifying the Fashion-MNIST dataset, there was a significant discrepancy in accuracy when classifying the CIFAR 10 dataset. The results indicate that the neural network model has an advantage in processing more complex datasets. However, the accuracy of the NN model in the second task was only 0.7157, partly due to its relatively simple structure of only two fully connected layers. In subsequent experiments, I will attempt to use a convolutional neural network (CNN) model, which typically outperforms non-convolutional neural networks when processing spatial data such as images and videos, as CNNs possess automatic feature extraction and learning capabilities for image features. Secondly, the data preprocessing module needs further improvement. In adjusting the model, I attempted to use data augmentation, but the accuracy was lower than before. In future experiments, I will continue to explore this issue.

7 References

- [1] Fukushima, Kunihiko. "Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position." *Biological cybernetics* 36 (BIOL), 1980, pp. 193-202
- [2] Hecht-Nielsen, Robert. "Theory of the backpropagation neural network." *Neural networks for perception*. Academic Press, 1992, pp. 65-93
- [3] Breiman L. "Random forests". *Machine learning (ML)*, 2001, pp. 5-32
- [4] Abadi, Martín, et al. "Tensorflow: a system for large-scale machine learning." *Osdi*. Vol. 16. No. 2016
- [5] B. Samia, Z. Soraya and M. Malika, "Fashion Images Classification using Machine Learning, Deep Learning and Transfer Learning Models," 2022 7th International Conference on Image and Signal Processing and their Applications (ISPA), Mostaganem, Algeria, 2022, pp. 1-5
- [6] F. Ertam and G. Aydın, "Data classification with deep learning using Tensorflow," 2017 International Conference on Computer Science and Engineering (UBMK), Antalya, Turkey, 2017, pp. 755-758
- [7] S. Huang, "Influence of Different Convolutional Neural Network Settings on the Performance of MNIST Handwritten Digits Recognition," 2020 International Conference on Artificial Intelligence and Education (ICAIE), Tianjin, China, 2020, pp. 1-6
- [8] Z. Zeng, Q. Gong and J. Zhang, "CNN Model Design of Gesture Recognition Based on Tensorflow Framework," 2019 IEEE 3rd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC), Chengdu, China, 2019, pp. 1062-1067
- [9] Shaik, Anjaneyulu Babu, and Sujatha Srinivasan. "A brief survey on random forest ensembles in classification model." *International Conference on Innovative Computing and Communications: Proceedings of ICICC 2018*, Volume 2. Springer Singapore, 2019, pp. 253-260

FINAL GRADE

77 / 100

GENERAL COMMENTS

Instructor

Abstract good. Well done. You can include summary results here too.

Review of lit/related work identifies relevant work and refs are of a good standard, using the school's agreed ref std. Good.

A well structured report and in general it is also well written. Well done.

The methodology and experiment are clearly presented and discussions of parameter choices and experiments are justified and argued for well.

More on dimensionality reduction and cross validation please.

Try to help the reader by including bar graphs as well as tables to summarise and compare results.

A good analysis of mis-classifications.

Overall a very good report. Well done,

PAGE 1

PAGE 2

PAGE 3

PAGE 4

PAGE 5

PAGE 6

PAGE 7

PAGE 8

PAGE 9

PAGE 10

PAGE 11

PAGE 12

PAGE 13

 **Comment 1**
it

PAGE 14

PAGE 15

GRADING FORM: SMU GENERIC_GJ

YIFEI SUN

0

RESEARCH-INFORMED LITERATURE

KNOWLEDGE AND UNDERSTANDING

ANALYSIS

PRACTICAL APPLICATION AND DEP.

SKILLS FOR PROFESSIONAL PRAC.