DaSilva_Bailleul_Classification_Nombres_Rapport_DeepLearning_E4FI

May 9, 2023

1 Présentation des expériences

Dans ces analyses, nous avons étudié la performance de différents types de réseaux de neurones pour la classification de chiffres écrits à la main. Nous avons commencé par l'implémentation d'un réseau de neurones simple couche, puis avons ajouté des couches cachées pour étudier l'impact sur les performances. Nous avons également exploré un réseau de neurones multicouches avec six couches pour voir si cela améliore encore davantage les résultats. À travers ces expériences, nous avons étudié l'impact du nombre de couches et de neurones sur les performances des réseaux de neurones pour la classification de chiffres manuscrits.

1.1 Réseaux de neurones simple couche

Nous réalisions l'implémentation d'un réseau de neurones simple couche à l'aide de la classe Network 2. Le but de l'expérience est d'entraîner un réseau de neurones simple couche afin d'en tirer sa performence.

Le modèle a été entraîné sur 30 epochs composé de 10 minibatchs chacune. Le coût sur les données d'entraînement a diminué progressivement au fil des epochs, passant de 0.731 à 0.662. Pendant ce temps, la précision du modèle sur les données d'entraînement a augmenté, passant de 90,4% à 92,0%. Cependant, la précision sur les données d'évaluation a oscillé entre 91,4% et 92,3%, sans aucune tendance significative à la hausse ou à la baisse.

Le modèle semble avoir atteint une précision d'environ 92% sur les données d'évaluation. Bien qu'il n'y ait pas eu de forte amélioration de la précision au fil des epochs, le modèle a réussi à éviter le surapprentissage et à maintenir une précision élevée sur les données d'évaluation.

1.2 Réseaux de neurones double couche

1.2.1 Couche cachée de 30 neurones

Il semble que l'ajout d'une couche cachée de 30 neurones ait permis d'améliorer les performances du réseau de neurones par rapport au précédent. L'exactitude sur l'ensemble d'évaluation est passée de 91,91 % à 96,17 %. Cependant, le coût d'évaluation a légèrement augmenté, passant de 0,526 à 0,756. Cela peut être dû à un surajustement du modèle, car l'exactitude sur l'ensemble d'entraînement est presque parfaite à la fin de l'entraînement, tandis que l'exactitude sur l'ensemble d'évaluation a cessé d'augmenter après environ 8 époques.

Il est également intéressant de noter que le coût d'évaluation a augmenté de manière constante après la quatrième époque, tandis que l'exactitude d'évaluation a stagné. Cela peut indiquer que le modèle commence à avoir des difficultés à généraliser les données et que des techniques telles que la régularisation peuvent être nécessaires pour améliorer les performances.

1.2.2 Couche cachée de 50 neurones

Pour le modèle à double couche cachée de 50 neurones, on peut observer que la précision sur les données d'apprentissage commence à atteindre des valeurs très élevées dès le premier epoch, et continue à augmenter graduellement jusqu'à presque atteindre 98% sur les données d'apprentissage après le 20e epoch. En ce qui concerne les données d'évaluation, on observe une précision qui commence à des valeurs plus faibles, mais qui augmente de manière constante au fil des epochs pour atteindre environ 97% sur les données d'évaluation après le 20e epoch. Cela suggère que le modèle est capable de généraliser des données qu'il n'a pas encore vues et qu'il ne souffre pas de sur-apprentissage.

Cependant, on peut observer que la perte sur les données d'apprentissage et d'évaluation continue à augmenter après environ 8 epochs, ce qui peut suggérer que le modèle a atteint son maximum de performance sur ces données et qu'il est peu probable qu'il améliore davantage son accuracy. Dans l'ensemble, le modèle semble avoir une performance solide sur cette tâche de classification de chiffres écrits à la main.

Si nous comparons la version avec 30 neurones en couche cachée avec la version avec 50 neurones en couche cachée, nous avons remarqué deux différence. Tout d'abord la version avec 30 neurones est moins précise que la version avec 50 neurones. Ensuite la version avec 30 neurones est légèrement plus rapide à s'entraîner que la version avec 50 neurones.

1.3 Réseaux de neurones multi couches

1.3.1 Réseau avec 6 couches

L'expérience est une implémentation d'un réseau de neurones multicouches avec 6 couches. Le nombre de neurones dans les couches est respectivement [784, 30, 30, 30, 30, 10]. La première couche a 784 nœuds, la couche de sortie a 10 nœuds car il y a 10 classes d'images à prédire (chiffres de 0 à 9) et les couches cachées ont 30 nœuds chacune.

Les résultats montrent une amélioration progressive des performances du modèle à chaque époque. Au départ, l'exactitude est très faible (~ 10%) et le coût est élevé. Au fur et à mesure que le modèle s'entraîne, l'exactitude augmente et le coût diminue. À la fin de la formation, le modèle atteint une précision d'environ 97% sur l'ensemble de données d'évaluation, ce qui est bien meilleur qu'avec un réseau à simple couche. Cependant, il est possible d'améliorer le résultat entre 2 à 20% si nous changeons le taux d'apprentissage de 0.1 à 0.03, mais il est important de notifier que celà à l'air de ralentir de le réseau et ne semble pas optimisé.

1.3.2 Réseau avec 12 couches

Le modèle de réseau de neurones avec 12 couches et les couches [784, 30, 30, 30, 30, 30, 30, 30, 30, 30, 30, 10] présente des signes de surapprentissage, car les résultats de l'entraînement et de l'évaluation indiquent que la précision sur l'ensemble d'entraînement est très élevée (plus de 50 000 images correctement classées sur 50 000), tandis que la précision sur l'ensemble d'évaluation est très faible (environ 1 000 images correctement classées sur 10 000). En d'autres termes, le modèle a réussi à apprendre à reconnaître parfaitement les images d'entraînement, mais il a échoué à généraliser à de nouvelles images. Il est possible que la complexité du modèle soit trop élevée pour le nombre limité de données d'entraînement disponibles, ce qui a conduit à un surapprentissage.

Pour améliorer les performances de ce modèle, des techniques de régularisation telles que la régularisation Lambda, l'abandon ou la normalisation des lots peuvent être utilisées pour éviter le surapprentissage. En outre, la réduction du nombre de couches ou des neurones dans chaque couche peut également améliorer les performances du modèle. Cependant, l'augmentation du nombre de couches peut présenter des limites, notamment en termes de complexité, de vanishing gradient et de temps de calcul plus long. De plus, l'augmentation du nombre de couches peut nécessiter un ensemble de données plus grand pour fournir suffisamment d'informations pour l'entraînement, ce qui peut affecter les performances du réseau si le nombre de données est limité.

1.4 Réseaux de neurones avec couches de convolution

Le réseau de neurones a la configuration suivante :

- une couche de convolution avec 20 filtres de 5x5, une fonction d'activation ReLU et une fenêtre de pooling de 2x2 ;
- une deuxième couche de convolution avec 40 filtres de 5x5, une fonction d'activation ReLU et une fenêtre de pooling de 2x2 ;
- une couche entièrement connectée avec 100 neurones et une fonction d'activation ReLU ;
- une couche de sortie softmax avec 10 neurones correspondant aux 10 classes de chiffres à reconnaître.

Le réseau est entraîné avec l'algorithme de descente de gradient stochastique (SGD) pour 30 epochs avec un taux d'apprentissage de 0,03 et une taille de mini-batch de 10. L'expérience utilise également une régularisation L2 avec un paramètre lambda de 0,1.

Les résultats de l'expérience sont donnés sous forme de précisions de validation et de test à chaque epoch. Les précisions augmentent régulièrement au cours des premiers epochs, atteignant une précision de validation maximale de 99.07% à l'epoch 7. Les précisions de test sont également très élevées, atteignant un maximum de 99.01%.

Comparé aux expériences précédentes réalisées avec un simple réseau de neurones multicouches, les résultats obtenus avec les couches de convolution sont bien meilleurs. En utilisant un réseau de neurones multicouches, la précision maximale de validation était d'environ 96%, tandis qu'avec les couches de convolution, la précision maximale atteinte est de 99.08%. Les couches de convolution permettent au réseau de mieux capturer les motifs spatiaux dans les images, ce qui est particulièrement important pour la reconnaissance d'images comme le MNIST dataset.

2 Annexe

2.1 Fonctions utilitaires

[1]:
```
def train_network(training_data, evaluation_data, layers=[784, 30, 10],␣
↪epochs=30):
    net = network2.Network(layers, cost=network2.CrossEntropyCost)
    evaluation_cost, evaluation_accuracy, training_cost, training_accuracy =␣
↪net.SGD(
        training_data,
        epochs=epochs,
        mini_batch_size=10,
        eta=0.1,
        lmbda=5.0,
        evaluation_data=evaluation_data,
        monitor_evaluation_cost=True,
        monitor_evaluation_accuracy=True,
        monitor_training_cost=True,
        monitor_training_accuracy=True
    )
    return net, evaluation_cost, evaluation_accuracy, training_cost,␣
↪training_accuracy
```

[2]:
```
def plot_predict(test_data):
    test_image = test_data[0][0]
    test_label = test_data[0][1]
    prediction = net.feedforward(test_image)
    predicted_label = np.argmax(prediction)

    plt.imshow(test_image.reshape(28, 28), cmap='gray')
    plt.title('Predicted label: {0}'.format(predicted_label))
    plt.show()
```

[3]:
```
def plot_conf_matrix(test_data, plot_title="Matrice de confusion Perceptron␣
↪\"de base\""):
    test_results = [(np.argmax(net.feedforward(image)), label) for (image,␣
↪label) in test_data]
    predictions = [result[0] for result in test_results]
    labels = [result[1] for result in test_results]

    conf_mat = confusion_matrix(labels, predictions)
    plt.imshow(conf_mat, cmap=plt.cm.Blues)
    plt.xlabel("Labels predits")
    plt.ylabel("Labels veridicts")
    plt.title(plot_title)
    plt.show()
```

[4]:
```
def subplots_data(data,  nb_plot_in_x=1, nb_plot_in_y=2):
    fig, axs = plt.subplots(nb_plot_in_x, nb_plot_in_y, figsize=(15, 10))

    for i, plot_data in enumerate(data):
        plot_idx_x = i // nb_plot_in_y
        plot_idx_y = i % nb_plot_in_y

        ax = axes[plot_idx_x][plot_idx_y] if nb_plot_in_x > 1 else␣
 ↪axs[plot_idx_y]

        ax.plot(plot_data['data'])
        ax.set_title(plot_data['plot_tile'])
        ax.set_xlabel(plot_data['label'][0])
        ax.set_ylabel(plot_data['label'][1])
    plt.show()
```

2.2 Expérimentations

2.2.1 Réseaux de neurones simple couche

[5]:
```
import network2
import network
import numpy as np
import mnist_loader
import matplotlib.pyplot as plt
from sklearn.metrics import confusion_matrix
```

[6]:
```
training_data, validation_data, test_data = mnist_loader.load_data_wrapper()
training_data = list(training_data)
```

[7]:
```
net, evaluation_cost, evaluation_accuracy, training_cost, training_accuracy =␣
 ↪train_network(
    training_data,
    validation_data,
    [784, 10]
)
```

```
Epoch 0 training complete
Cost on training data: 0.770620811706
Accuracy on training data: 45031 / 50000
Cost on evaluation data: 0.75532336307
Accuracy on evaluation data: 9091 / 10000

Epoch 1 training complete
Cost on training data: 0.721677538675
Accuracy on training data: 45444 / 50000
```

```
Cost on evaluation data: 0.738123771745
Accuracy on evaluation data: 9146 / 10000

Epoch 2 training complete
Cost on training data: 0.694002500336
Accuracy on training data: 45602 / 50000
Cost on evaluation data: 0.723168603752
Accuracy on evaluation data: 9184 / 10000

Epoch 3 training complete
Cost on training data: 0.737426801676
Accuracy on training data: 45350 / 50000
Cost on evaluation data: 0.782477335715
Accuracy on evaluation data: 9120 / 10000

Epoch 4 training complete
Cost on training data: 0.700827870724
Accuracy on training data: 45625 / 50000
Cost on evaluation data: 0.753416665362
Accuracy on evaluation data: 9171 / 10000

Epoch 5 training complete
Cost on training data: 0.706141588828
Accuracy on training data: 45731 / 50000
Cost on evaluation data: 0.761386672188
Accuracy on evaluation data: 9198 / 10000

Epoch 6 training complete
Cost on training data: 0.683439207473
Accuracy on training data: 45773 / 50000
Cost on evaluation data: 0.750193555537
Accuracy on evaluation data: 9185 / 10000

Epoch 7 training complete
Cost on training data: 0.693856705551
Accuracy on training data: 45711 / 50000
Cost on evaluation data: 0.765211044432
Accuracy on evaluation data: 9155 / 10000

Epoch 8 training complete
Cost on training data: 0.688092102697
Accuracy on training data: 45622 / 50000
Cost on evaluation data: 0.767693743986
Accuracy on evaluation data: 9169 / 10000

Epoch 9 training complete
Cost on training data: 0.683590744321
Accuracy on training data: 45797 / 50000
```

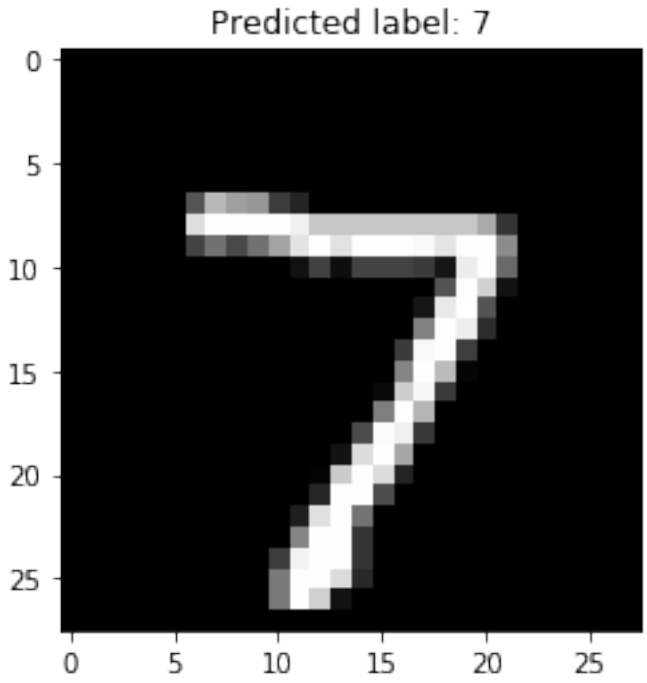
```
Cost on evaluation data: 0.763461039728
Accuracy on evaluation data: 9204 / 10000

Epoch 10 training complete
Cost on training data: 0.679562503186
Accuracy on training data: 45751 / 50000
Cost on evaluation data: 0.772220428148
Accuracy on evaluation data: 9181 / 10000

Epoch 11 training complete
Cost on training data: 0.675578297031
Accuracy on training data: 45833 / 50000
Cost on evaluation data: 0.772088014215
Accuracy on evaluation data: 9189 / 10000

Epoch 12 training complete
Cost on training data: 0.69515486037
Accuracy on training data: 45752 / 50000
Cost on evaluation data: 0.790353556204
Accuracy on evaluation data: 9149 / 10000

Epoch 13 training complete
Cost on training data: 0.681771920909
Accuracy on training data: 45864 / 50000
Cost on evaluation data: 0.786192287249
Accuracy on evaluation data: 9191 / 10000

Epoch 14 training complete
Cost on training data: 0.673775988193
Accuracy on training data: 45838 / 50000
Cost on evaluation data: 0.773219288483
Accuracy on evaluation data: 9177 / 10000

Epoch 15 training complete
Cost on training data: 0.674231016771
Accuracy on training data: 45885 / 50000
Cost on evaluation data: 0.780700791974
Accuracy on evaluation data: 9193 / 10000

Epoch 16 training complete
Cost on training data: 0.670515555695
Accuracy on training data: 45993 / 50000
Cost on evaluation data: 0.776541013918
Accuracy on evaluation data: 9202 / 10000

Epoch 17 training complete
Cost on training data: 0.678358839565
Accuracy on training data: 45868 / 50000
```

```
Cost on evaluation data: 0.787711093028
Accuracy on evaluation data: 9190 / 10000

Epoch 18 training complete
Cost on training data: 0.687310676535
Accuracy on training data: 45754 / 50000
Cost on evaluation data: 0.799646836256
Accuracy on evaluation data: 9190 / 10000

Epoch 19 training complete
Cost on training data: 0.681108140522
Accuracy on training data: 45880 / 50000
Cost on evaluation data: 0.795029171716
Accuracy on evaluation data: 9196 / 10000

Epoch 20 training complete
Cost on training data: 0.701693215671
Accuracy on training data: 45734 / 50000
Cost on evaluation data: 0.81837987513
Accuracy on evaluation data: 9193 / 10000

Epoch 21 training complete
Cost on training data: 0.671256774557
Accuracy on training data: 45971 / 50000
Cost on evaluation data: 0.789579874713
Accuracy on evaluation data: 9200 / 10000

Epoch 22 training complete
Cost on training data: 0.66719307377
Accuracy on training data: 45992 / 50000
Cost on evaluation data: 0.787115731235
Accuracy on evaluation data: 9210 / 10000

Epoch 23 training complete
Cost on training data: 0.677558916431
Accuracy on training data: 45906 / 50000
Cost on evaluation data: 0.796650181818
Accuracy on evaluation data: 9202 / 10000

Epoch 24 training complete
Cost on training data: 0.713932278136
Accuracy on training data: 45704 / 50000
Cost on evaluation data: 0.824684863502
Accuracy on evaluation data: 9158 / 10000

Epoch 25 training complete
Cost on training data: 0.667652296108
Accuracy on training data: 45983 / 50000
```

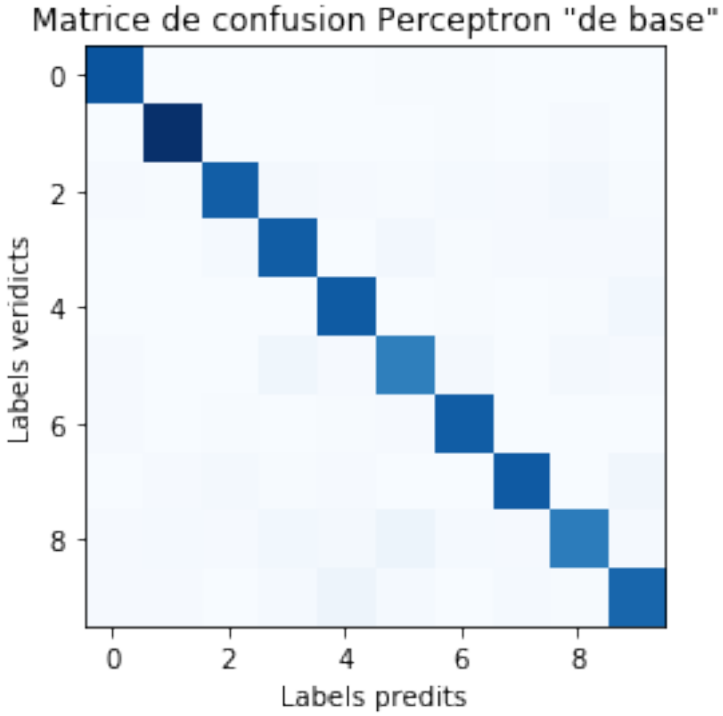
```
Cost on evaluation data: 0.793917039601
Accuracy on evaluation data: 9204 / 10000

Epoch 26 training complete
Cost on training data: 0.669767225922
Accuracy on training data: 45954 / 50000
Cost on evaluation data: 0.792370214083
Accuracy on evaluation data: 9182 / 10000

Epoch 27 training complete
Cost on training data: 0.674967163707
Accuracy on training data: 45920 / 50000
Cost on evaluation data: 0.794061087924
Accuracy on evaluation data: 9208 / 10000

Epoch 28 training complete
Cost on training data: 0.670030908529
Accuracy on training data: 45963 / 50000
Cost on evaluation data: 0.794719471964
Accuracy on evaluation data: 9204 / 10000

Epoch 29 training complete
Cost on training data: 0.680401449617
Accuracy on training data: 45823 / 50000
Cost on evaluation data: 0.807514260202
Accuracy on evaluation data: 9183 / 10000
```

[8]:
```
validation_data = list(validation_data)
test_data = list(test_data)
```

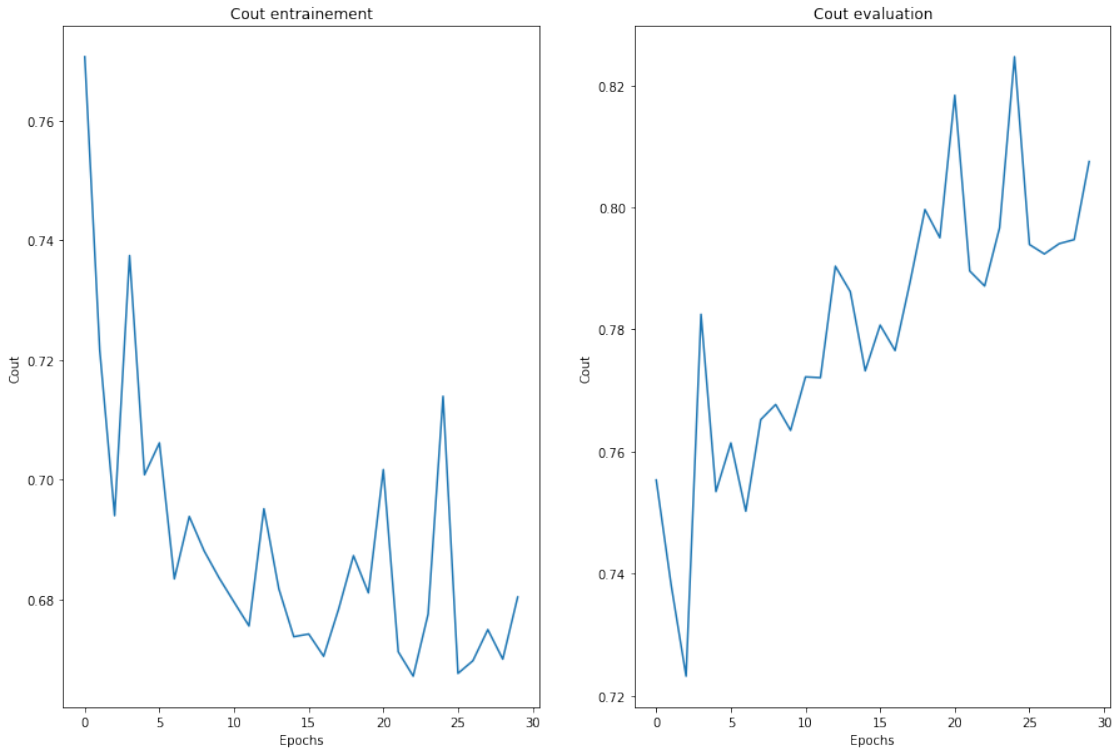
[9]:
```
plot_predict(test_data)
```



[10]:
```
plot_conf_matrix(test_data)
```

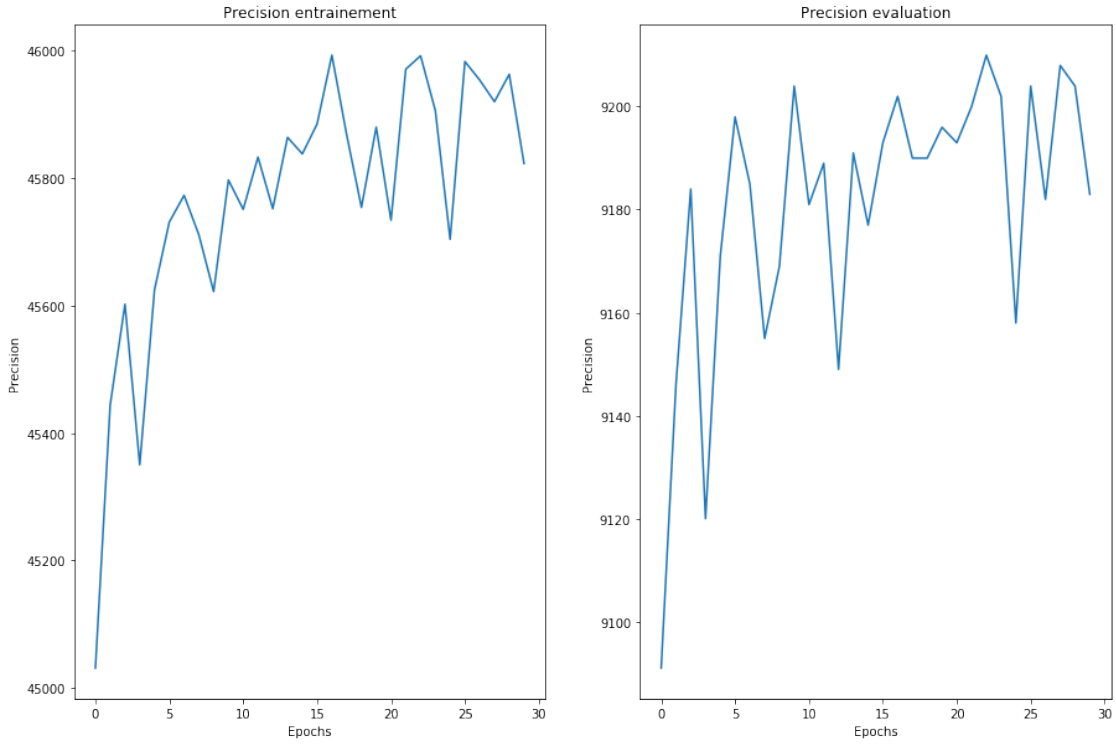


[11]:
```
subplots_data(
    [
        {
            "data": training_cost,
            "plot_tile": 'Cout entrainement',
            "label": ('Epochs', 'Cout'),
        },
        {
            "data": evaluation_cost,
            "plot_tile": 'Cout evaluation',
            "label": ('Epochs', 'Cout'),
        }
    ],
    1,
    2
)
```



[12]:
```
subplots_data(
    [
        {
            "data": training_accuracy,
            "plot_tile": 'Precision entrainement',
            "label": ('Epochs', 'Precision'),
```

```
        },
        {
            "data": evaluation_accuracy,
            "plot_tile": 'Precision evaluation',
            "label": ('Epochs', 'Precision'),
        }
    ],
    1,
    2
)
```



2.2.2 Réseaux de neurones double couches

[13]:
```
import network2
import numpy as np
import mnist_loader
import matplotlib.pyplot as plt
from sklearn.metrics import confusion_matrix
```

Couche cachée de 30 neurones

[14]:
```
training_data, validation_data, test_data = mnist_loader.load_data_wrapper()
training_data = list(training_data)
```

[15]:
```
net, evaluation_cost, evaluation_accuracy, training_cost, training_accuracy =␣
↪train_network(
    training_data,
    validation_data,
    [784, 30, 10]
)
```

```
Epoch 0 training complete
Cost on training data: 0.5549886333
Accuracy on training data: 46208 / 50000
Cost on evaluation data: 0.65362663042
Accuracy on evaluation data: 9273 / 10000

Epoch 1 training complete
Cost on training data: 0.46416862591
Accuracy on training data: 46998 / 50000
Cost on evaluation data: 0.629189008308
Accuracy on evaluation data: 9419 / 10000

Epoch 2 training complete
Cost on training data: 0.424339164666
Accuracy on training data: 47361 / 50000
Cost on evaluation data: 0.634017516037
Accuracy on evaluation data: 9477 / 10000

Epoch 3 training complete
Cost on training data: 0.404672020626
Accuracy on training data: 47600 / 50000
Cost on evaluation data: 0.651482594235
Accuracy on evaluation data: 9506 / 10000

Epoch 4 training complete
Cost on training data: 0.383352493519
Accuracy on training data: 47825 / 50000
Cost on evaluation data: 0.657740565717
Accuracy on evaluation data: 9537 / 10000

Epoch 5 training complete
Cost on training data: 0.372696444782
Accuracy on training data: 47915 / 50000
Cost on evaluation data: 0.670233169342
Accuracy on evaluation data: 9562 / 10000

Epoch 6 training complete
```

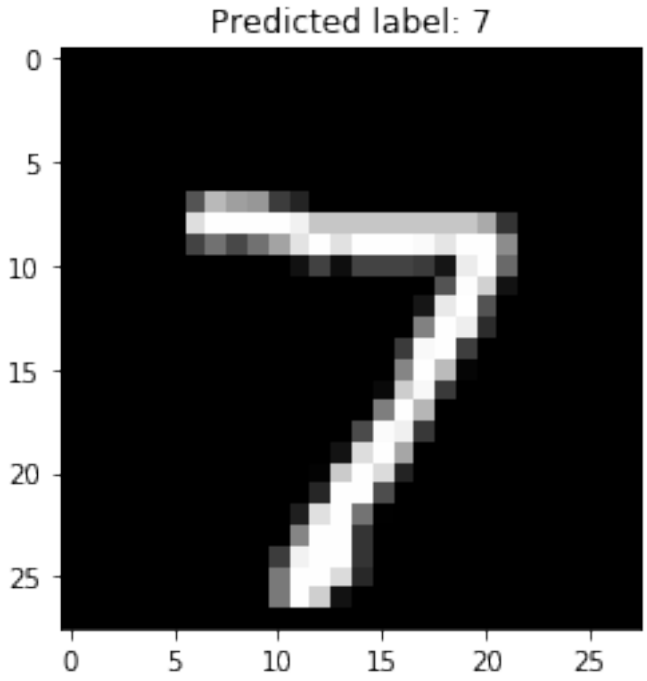
```
Cost on training data: 0.36208270955
Accuracy on training data: 48043 / 50000
Cost on evaluation data: 0.682763463264
Accuracy on evaluation data: 9558 / 10000

Epoch 7 training complete
Cost on training data: 0.353427278129
Accuracy on training data: 48120 / 50000
Cost on evaluation data: 0.692718404648
Accuracy on evaluation data: 9588 / 10000

Epoch 8 training complete
Cost on training data: 0.348026785453
Accuracy on training data: 48202 / 50000
Cost on evaluation data: 0.700859495247
Accuracy on evaluation data: 9582 / 10000

Epoch 9 training complete
Cost on training data: 0.342537726262
Accuracy on training data: 48324 / 50000
Cost on evaluation data: 0.707958397224
Accuracy on evaluation data: 9608 / 10000

Epoch 10 training complete
Cost on training data: 0.33992264538
Accuracy on training data: 48368 / 50000
Cost on evaluation data: 0.723730452609
Accuracy on evaluation data: 9604 / 10000

Epoch 11 training complete
Cost on training data: 0.332912860338
Accuracy on training data: 48463 / 50000
Cost on evaluation data: 0.727738473399
Accuracy on evaluation data: 9620 / 10000

Epoch 12 training complete
Cost on training data: 0.325359899179
Accuracy on training data: 48511 / 50000
Cost on evaluation data: 0.72904093
Accuracy on evaluation data: 9620 / 10000

Epoch 13 training complete
Cost on training data: 0.328641887837
Accuracy on training data: 48568 / 50000
Cost on evaluation data: 0.74042124326
Accuracy on evaluation data: 9616 / 10000

Epoch 14 training complete
```

```
Cost on training data: 0.320542861706
Accuracy on training data: 48546 / 50000
Cost on evaluation data: 0.74569908819
Accuracy on evaluation data: 9640 / 10000

Epoch 15 training complete
Cost on training data: 0.322980153351
Accuracy on training data: 48572 / 50000
Cost on evaluation data: 0.758969167954
Accuracy on evaluation data: 9632 / 10000

Epoch 16 training complete
Cost on training data: 0.317920926058
Accuracy on training data: 48627 / 50000
Cost on evaluation data: 0.757694636921
Accuracy on evaluation data: 9637 / 10000

Epoch 17 training complete
Cost on training data: 0.314586212876
Accuracy on training data: 48666 / 50000
Cost on evaluation data: 0.765428689831
Accuracy on evaluation data: 9645 / 10000

Epoch 18 training complete
Cost on training data: 0.313454704723
Accuracy on training data: 48696 / 50000
Cost on evaluation data: 0.764355415116
Accuracy on evaluation data: 9663 / 10000

Epoch 19 training complete
Cost on training data: 0.309132710143
Accuracy on training data: 48760 / 50000
Cost on evaluation data: 0.767739208059
Accuracy on evaluation data: 9647 / 10000

Epoch 20 training complete
Cost on training data: 0.31174635461
Accuracy on training data: 48756 / 50000
Cost on evaluation data: 0.777908613189
Accuracy on evaluation data: 9647 / 10000

Epoch 21 training complete
Cost on training data: 0.313299602668
Accuracy on training data: 48746 / 50000
Cost on evaluation data: 0.781664320304
Accuracy on evaluation data: 9649 / 10000

Epoch 22 training complete
```

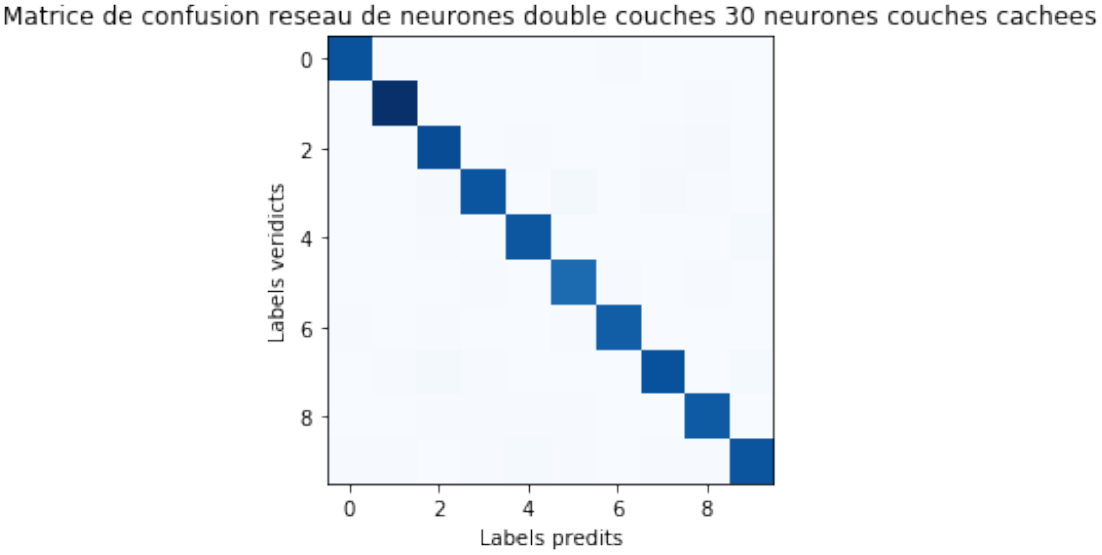
```
Cost on training data: 0.311686354153
Accuracy on training data: 48732 / 50000
Cost on evaluation data: 0.786470532153
Accuracy on evaluation data: 9643 / 10000

Epoch 23 training complete
Cost on training data: 0.309493519697
Accuracy on training data: 48783 / 50000
Cost on evaluation data: 0.788728722972
Accuracy on evaluation data: 9654 / 10000

Epoch 24 training complete
Cost on training data: 0.309809981791
Accuracy on training data: 48779 / 50000
Cost on evaluation data: 0.789114371989
Accuracy on evaluation data: 9656 / 10000

Epoch 25 training complete
Cost on training data: 0.30395361628
Accuracy on training data: 48835 / 50000
Cost on evaluation data: 0.791734225691
Accuracy on evaluation data: 9656 / 10000

Epoch 26 training complete
Cost on training data: 0.308656209292
Accuracy on training data: 48772 / 50000
Cost on evaluation data: 0.801440450707
Accuracy on evaluation data: 9649 / 10000

Epoch 27 training complete
Cost on training data: 0.311215785102
Accuracy on training data: 48751 / 50000
Cost on evaluation data: 0.802094837896
Accuracy on evaluation data: 9637 / 10000

Epoch 28 training complete
Cost on training data: 0.302559488613
Accuracy on training data: 48872 / 50000
Cost on evaluation data: 0.795252203318
Accuracy on evaluation data: 9668 / 10000

Epoch 29 training complete
Cost on training data: 0.303924227251
Accuracy on training data: 48845 / 50000
Cost on evaluation data: 0.799390314688
Accuracy on evaluation data: 9664 / 10000
```

[16]:
```
validation_data = list(validation_data)
test_data = list(test_data)
```

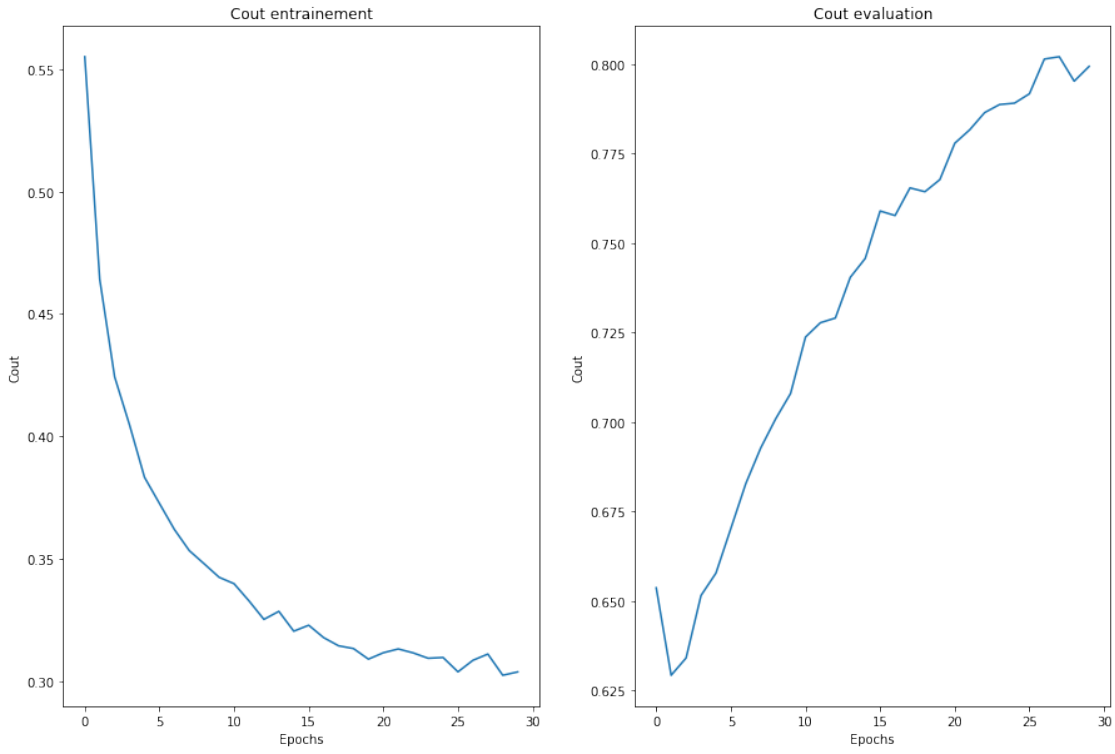
[17]:
```
plot_predict(test_data)
```



[18]:
```
plot_conf_matrix(test_data, plot_title="Matrice de confusion reseau de neurones␣
↪double couches 30 neurones couches cachees")
```

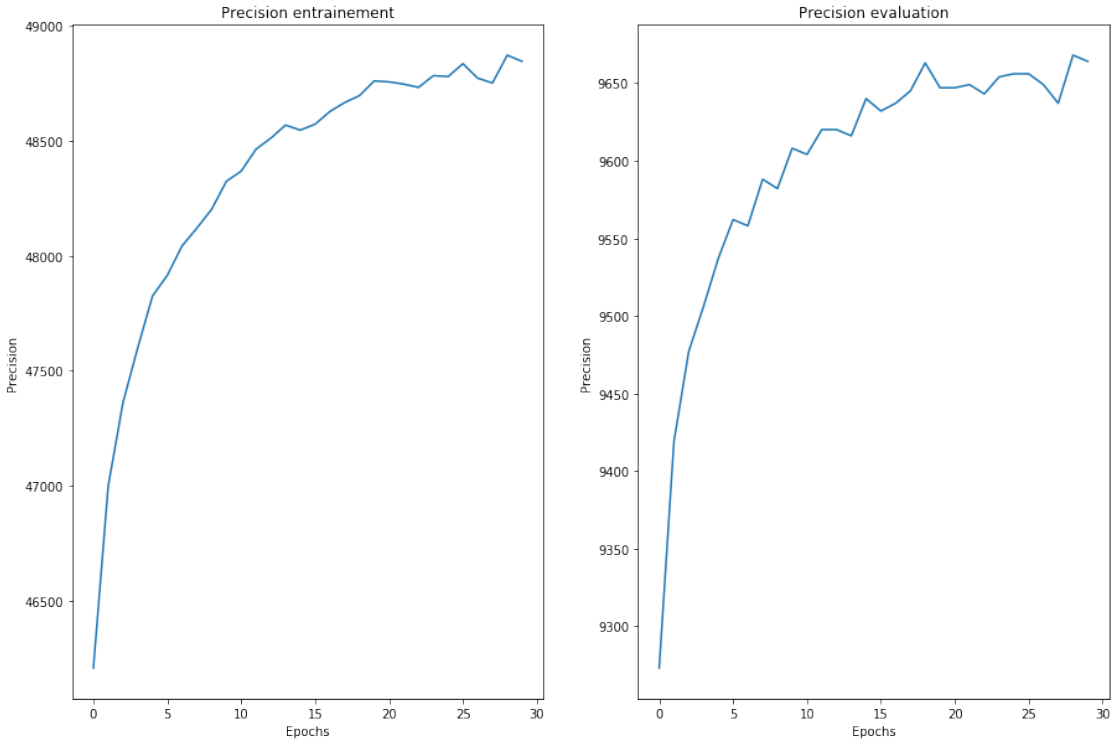


[19]:
```
subplots_data(
    [
        {
            "data": training_cost,
            "plot_tile": 'Cout entrainement',
            "label": ('Epochs', 'Cout'),
        },
        {
            "data": evaluation_cost,
            "plot_tile": 'Cout evaluation',
            "label": ('Epochs', 'Cout'),
        }
    ],
    1,
    2
)
```



[20]:
```
subplots_data(
    [
        {
            "data": training_accuracy,
```

```
            "plot_tile": 'Precision entrainement',
            "label": ('Epochs', 'Precision'),
        },
        {
            "data": evaluation_accuracy,
            "plot_tile": 'Precision evaluation',
            "label": ('Epochs', 'Precision'),
        }
    ],
    1,
    2
)
```



Couche cachée de 50 neurones

[21]:
```
training_data, validation_data, test_data = mnist_loader.load_data_wrapper()
training_data = list(training_data)
```

[22]:
```
net, evaluation_cost, evaluation_accuracy, training_cost, training_accuracy =␣
↪train_network(
    training_data,
    validation_data,
    [784, 50, 10]
```

```
)
```

```
Epoch 0 training complete
Cost on training data: 0.526146841155
Accuracy on training data: 46331 / 50000
Cost on evaluation data: 0.624190253452
Accuracy on evaluation data: 9317 / 10000

Epoch 1 training complete
Cost on training data: 0.434100680457
Accuracy on training data: 47168 / 50000
Cost on evaluation data: 0.595569285874
Accuracy on evaluation data: 9479 / 10000

Epoch 2 training complete
Cost on training data: 0.392223333571
Accuracy on training data: 47603 / 50000
Cost on evaluation data: 0.600366692281
Accuracy on evaluation data: 9546 / 10000

Epoch 3 training complete
Cost on training data: 0.359664907675
Accuracy on training data: 47898 / 50000
Cost on evaluation data: 0.607427897976
Accuracy on evaluation data: 9590 / 10000

Epoch 4 training complete
Cost on training data: 0.34389593484
Accuracy on training data: 48091 / 50000
Cost on evaluation data: 0.623714239363
Accuracy on evaluation data: 9602 / 10000

Epoch 5 training complete
Cost on training data: 0.330163264519
Accuracy on training data: 48269 / 50000
Cost on evaluation data: 0.637963479465
Accuracy on evaluation data: 9619 / 10000

Epoch 6 training complete
Cost on training data: 0.315201716126
Accuracy on training data: 48416 / 50000
Cost on evaluation data: 0.642913518502
Accuracy on evaluation data: 9640 / 10000

Epoch 7 training complete
Cost on training data: 0.306336236912
Accuracy on training data: 48525 / 50000
```

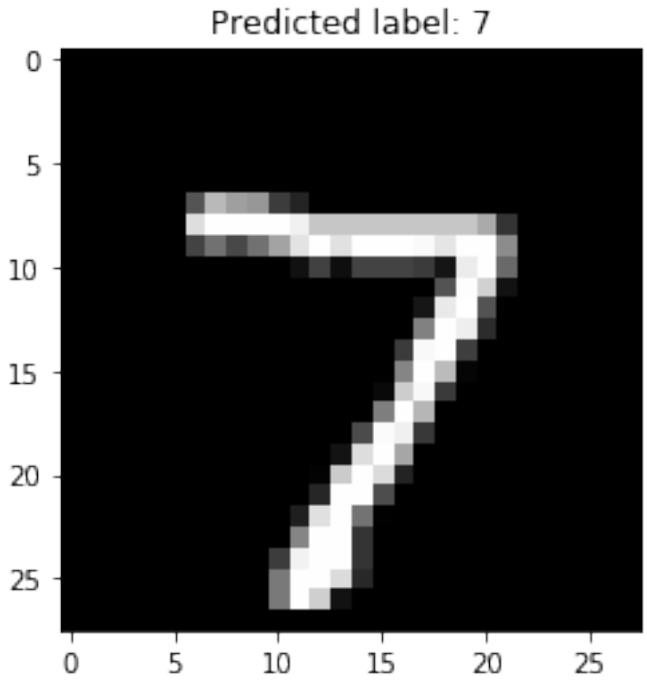
```
Cost on evaluation data: 0.655742055798
Accuracy on evaluation data: 9658 / 10000

Epoch 8 training complete
Cost on training data: 0.301866047837
Accuracy on training data: 48622 / 50000
Cost on evaluation data: 0.673695466889
Accuracy on evaluation data: 9664 / 10000

Epoch 9 training complete
Cost on training data: 0.306543152994
Accuracy on training data: 48613 / 50000
Cost on evaluation data: 0.694705055659
Accuracy on evaluation data: 9660 / 10000

Epoch 10 training complete
Cost on training data: 0.290740832222
Accuracy on training data: 48747 / 50000
Cost on evaluation data: 0.692121269545
Accuracy on evaluation data: 9679 / 10000

Epoch 11 training complete
Cost on training data: 0.291283273204
Accuracy on training data: 48775 / 50000
Cost on evaluation data: 0.703712376577
Accuracy on evaluation data: 9692 / 10000

Epoch 12 training complete
Cost on training data: 0.279907461165
Accuracy on training data: 48890 / 50000
Cost on evaluation data: 0.706813638521
Accuracy on evaluation data: 9685 / 10000

Epoch 13 training complete
Cost on training data: 0.274747471267
Accuracy on training data: 48934 / 50000
Cost on evaluation data: 0.710727061343
Accuracy on evaluation data: 9700 / 10000

Epoch 14 training complete
Cost on training data: 0.282487112664
Accuracy on training data: 48928 / 50000
Cost on evaluation data: 0.729260361003
Accuracy on evaluation data: 9682 / 10000

Epoch 15 training complete
Cost on training data: 0.272082402675
Accuracy on training data: 48981 / 50000
```

```
Cost on evaluation data: 0.727131037294
Accuracy on evaluation data: 9704 / 10000

Epoch 16 training complete
Cost on training data: 0.271647872884
Accuracy on training data: 49024 / 50000
Cost on evaluation data: 0.733435611049
Accuracy on evaluation data: 9701 / 10000

Epoch 17 training complete
Cost on training data: 0.270095112525
Accuracy on training data: 49072 / 50000
Cost on evaluation data: 0.738441086944
Accuracy on evaluation data: 9712 / 10000

Epoch 18 training complete
Cost on training data: 0.269013363518
Accuracy on training data: 49084 / 50000
Cost on evaluation data: 0.743634648132
Accuracy on evaluation data: 9702 / 10000

Epoch 19 training complete
Cost on training data: 0.26634765297
Accuracy on training data: 49104 / 50000
Cost on evaluation data: 0.747624212836
Accuracy on evaluation data: 9714 / 10000

Epoch 20 training complete
Cost on training data: 0.26524440907
Accuracy on training data: 49134 / 50000
Cost on evaluation data: 0.751121222756
Accuracy on evaluation data: 9726 / 10000

Epoch 21 training complete
Cost on training data: 0.263799632622
Accuracy on training data: 49179 / 50000
Cost on evaluation data: 0.751029452638
Accuracy on evaluation data: 9716 / 10000

Epoch 22 training complete
Cost on training data: 0.26267666212
Accuracy on training data: 49206 / 50000
Cost on evaluation data: 0.75529904548
Accuracy on evaluation data: 9725 / 10000

Epoch 23 training complete
Cost on training data: 0.265016121035
Accuracy on training data: 49157 / 50000
```

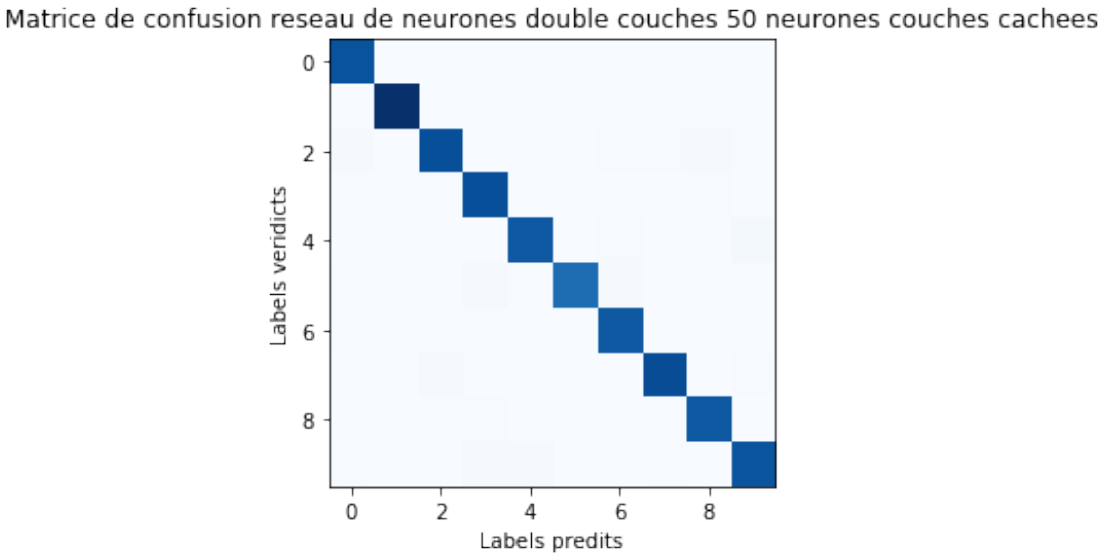
```
Cost on evaluation data: 0.764757175699
Accuracy on evaluation data: 9698 / 10000

Epoch 24 training complete
Cost on training data: 0.259050725574
Accuracy on training data: 49215 / 50000
Cost on evaluation data: 0.763109107918
Accuracy on evaluation data: 9716 / 10000

Epoch 25 training complete
Cost on training data: 0.264216666441
Accuracy on training data: 49193 / 50000
Cost on evaluation data: 0.775909468783
Accuracy on evaluation data: 9699 / 10000

Epoch 26 training complete
Cost on training data: 0.259273969885
Accuracy on training data: 49205 / 50000
Cost on evaluation data: 0.774526972093
Accuracy on evaluation data: 9713 / 10000

Epoch 27 training complete
Cost on training data: 0.257758557785
Accuracy on training data: 49227 / 50000
Cost on evaluation data: 0.770716791466
Accuracy on evaluation data: 9729 / 10000

Epoch 28 training complete
Cost on training data: 0.263201586795
Accuracy on training data: 49209 / 50000
Cost on evaluation data: 0.781090156315
Accuracy on evaluation data: 9709 / 10000

Epoch 29 training complete
Cost on training data: 0.257749657497
Accuracy on training data: 49264 / 50000
Cost on evaluation data: 0.778537355669
Accuracy on evaluation data: 9721 / 10000
```

[23]:
```
validation_data = list(validation_data)
test_data = list(test_data)
```

[24]:
```
plot_predict(test_data)
```

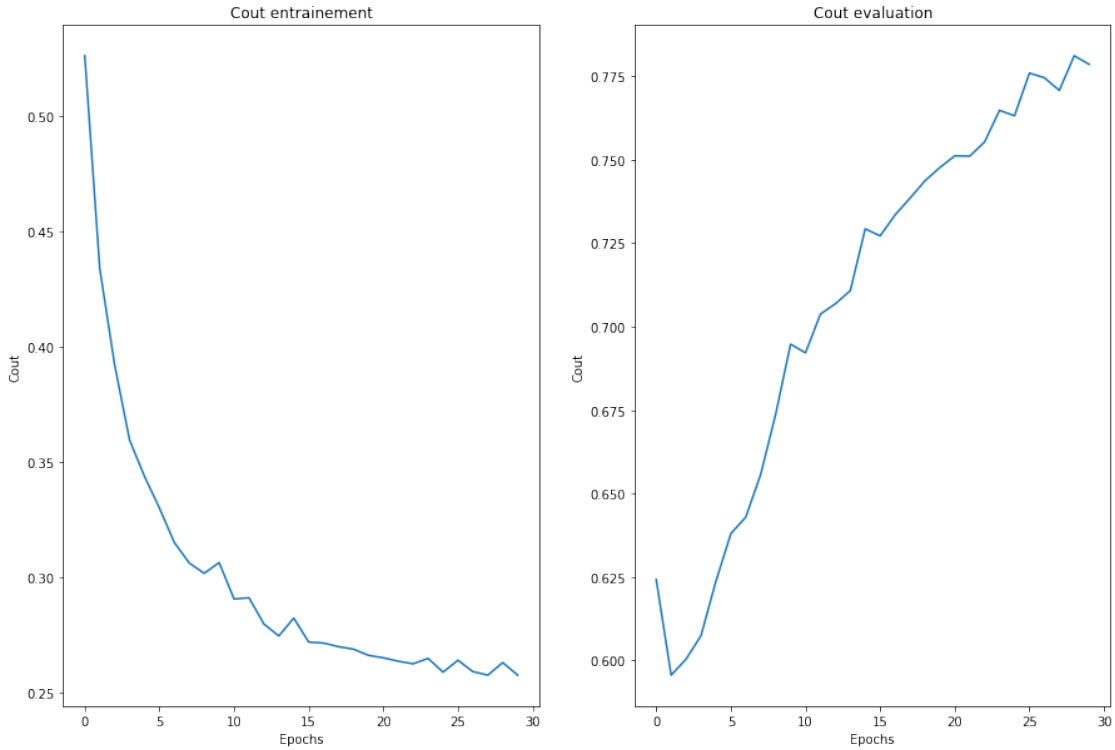


[25]:
```
plot_conf_matrix(test_data, plot_title="Matrice de confusion reseau de neurones␣
↪double couches 50 neurones couches cachees")
```



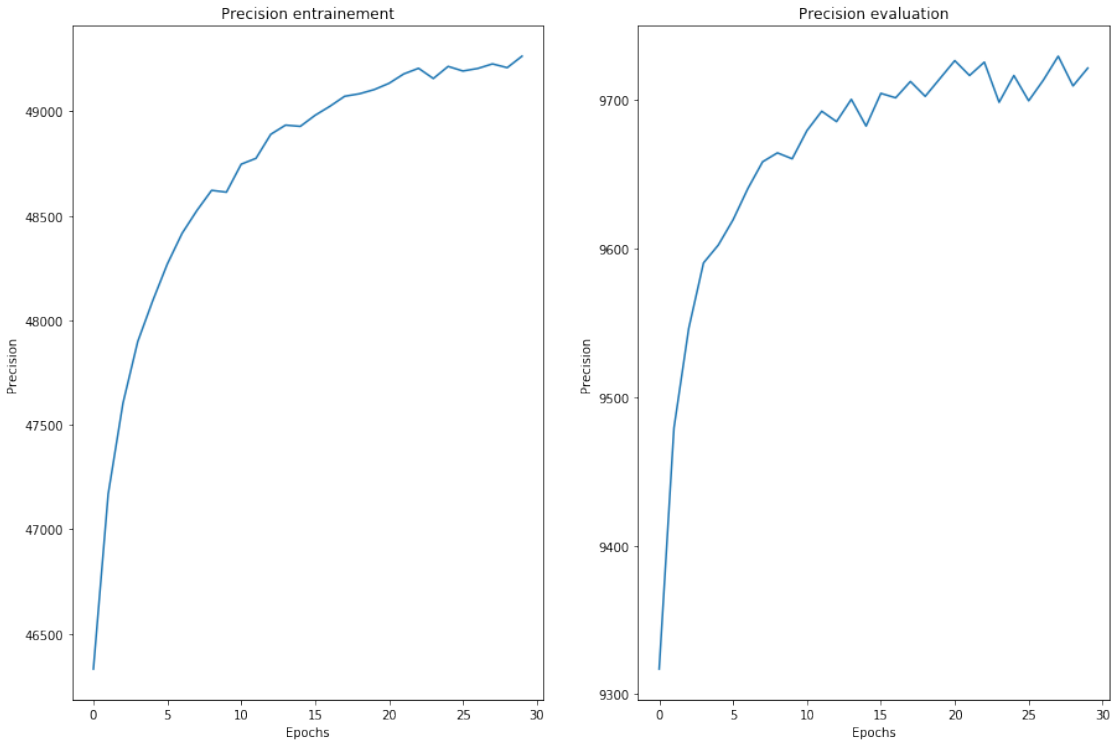
[26]:
```
subplots_data(
    [
        {
```

```
            "data": training_cost,
            "plot_tile": 'Cout entrainement',
            "label": ('Epochs', 'Cout'),
        },
        {
            "data": evaluation_cost,
            "plot_tile": 'Cout evaluation',
            "label": ('Epochs', 'Cout'),
        }
    ],
    1,
    2
)
```



[27]:
```
subplots_data(
    [
        {
            "data": training_accuracy,
            "plot_tile": 'Precision entrainement',
            "label": ('Epochs', 'Precision'),
        },
        {
            "data": evaluation_accuracy,
```

```
            "plot_tile": 'Precision evaluation',
            "label": ('Epochs', 'Precision'),
        }
    ],
    1,
    2
)
```



2.2.3 Réseaux de neurones multi couches

Réseau avec 6 couches

[28]:
```
import network2
import numpy as np
import mnist_loader
import matplotlib.pyplot as plt
from sklearn.metrics import confusion_matrix
```

[29]:
```
training_data, validation_data, test_data = mnist_loader.load_data_wrapper()
training_data = list(training_data)
```

[30]:
```
net, evaluation_cost, evaluation_accuracy, training_cost, training_accuracy =␣
 ↪train_network(
```

```
    training_data,
    validation_data,
    [784, 30, 30, 30, 30, 10]
)
```

```
Epoch 0 training complete
Cost on training data: 3.08557760446
Accuracy on training data: 10551 / 50000
Cost on evaluation data: 3.12020488863
Accuracy on evaluation data: 2156 / 10000

Epoch 1 training complete
Cost on training data: 1.92888168142
Accuracy on training data: 26227 / 50000
Cost on evaluation data: 2.03758742189
Accuracy on evaluation data: 5343 / 10000

Epoch 2 training complete
Cost on training data: 0.940258744629
Accuracy on training data: 43152 / 50000
Cost on evaluation data: 1.19764789842
Accuracy on evaluation data: 8699 / 10000

Epoch 3 training complete
Cost on training data: 0.598762652191
Accuracy on training data: 46326 / 50000
Cost on evaluation data: 0.956450220372
Accuracy on evaluation data: 9281 / 10000

Epoch 4 training complete
Cost on training data: 0.492580142785
Accuracy on training data: 47212 / 50000
Cost on evaluation data: 0.92741015386
Accuracy on evaluation data: 9386 / 10000

Epoch 5 training complete
Cost on training data: 0.485581181263
Accuracy on training data: 47276 / 50000
Cost on evaluation data: 0.962300441531
Accuracy on evaluation data: 9397 / 10000

Epoch 6 training complete
Cost on training data: 0.422887262745
Accuracy on training data: 47714 / 50000
Cost on evaluation data: 0.93325830453
Accuracy on evaluation data: 9488 / 10000
```

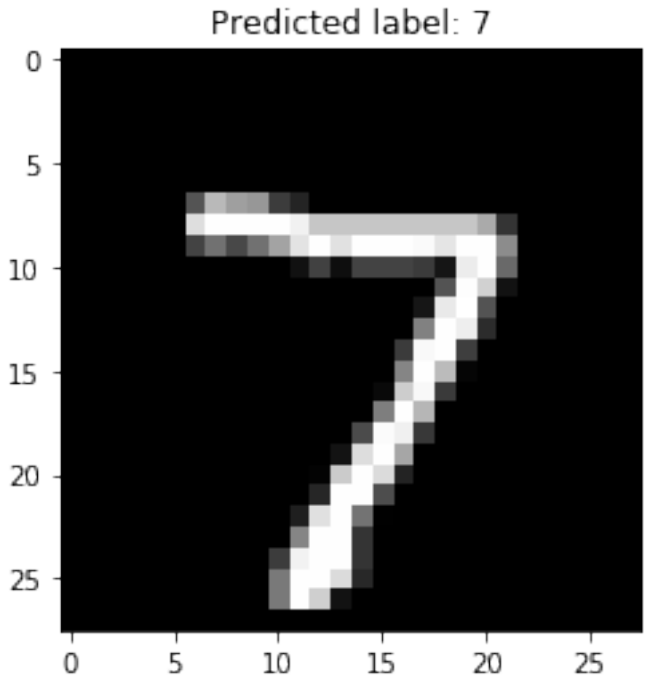
```
Epoch 7 training complete
Cost on training data: 0.417732626479
Accuracy on training data: 47748 / 50000
Cost on evaluation data: 0.950069434166
Accuracy on evaluation data: 9478 / 10000

Epoch 8 training complete
Cost on training data: 0.398148712829
Accuracy on training data: 47987 / 50000
Cost on evaluation data: 0.95767668442
Accuracy on evaluation data: 9509 / 10000

Epoch 9 training complete
Cost on training data: 0.373205111674
Accuracy on training data: 48145 / 50000
Cost on evaluation data: 0.940294175169
Accuracy on evaluation data: 9540 / 10000

Epoch 10 training complete
Cost on training data: 0.356151754574
Accuracy on training data: 48351 / 50000
Cost on evaluation data: 0.939085954267
Accuracy on evaluation data: 9555 / 10000

Epoch 11 training complete
Cost on training data: 0.328964042816
Accuracy on training data: 48606 / 50000
Cost on evaluation data: 0.93225204537
Accuracy on evaluation data: 9591 / 10000

Epoch 12 training complete
Cost on training data: 0.332173941664
Accuracy on training data: 48549 / 50000
Cost on evaluation data: 0.937722745572
Accuracy on evaluation data: 9582 / 10000

Epoch 13 training complete
Cost on training data: 0.339550971296
Accuracy on training data: 48480 / 50000
Cost on evaluation data: 0.968065639365
Accuracy on evaluation data: 9540 / 10000

Epoch 14 training complete
Cost on training data: 0.318530957901
Accuracy on training data: 48685 / 50000
Cost on evaluation data: 0.945976557324
Accuracy on evaluation data: 9581 / 10000
```

```
Epoch 15 training complete
Cost on training data: 0.323440927425
Accuracy on training data: 48629 / 50000
Cost on evaluation data: 0.95610076012
Accuracy on evaluation data: 9595 / 10000

Epoch 16 training complete
Cost on training data: 0.31819612659
Accuracy on training data: 48689 / 50000
Cost on evaluation data: 0.960382314422
Accuracy on evaluation data: 9585 / 10000

Epoch 17 training complete
Cost on training data: 0.31830858589
Accuracy on training data: 48673 / 50000
Cost on evaluation data: 0.956052466629
Accuracy on evaluation data: 9580 / 10000

Epoch 18 training complete
Cost on training data: 0.307422327226
Accuracy on training data: 48797 / 50000
Cost on evaluation data: 0.951928608295
Accuracy on evaluation data: 9596 / 10000

Epoch 19 training complete
Cost on training data: 0.289968918765
Accuracy on training data: 48906 / 50000
Cost on evaluation data: 0.95270449474
Accuracy on evaluation data: 9606 / 10000

Epoch 20 training complete
Cost on training data: 0.320493032007
Accuracy on training data: 48602 / 50000
Cost on evaluation data: 0.980065598593
Accuracy on evaluation data: 9569 / 10000

Epoch 21 training complete
Cost on training data: 0.28756416216
Accuracy on training data: 48969 / 50000
Cost on evaluation data: 0.956189980192
Accuracy on evaluation data: 9616 / 10000

Epoch 22 training complete
Cost on training data: 0.299777099347
Accuracy on training data: 48862 / 50000
Cost on evaluation data: 0.971658950849
Accuracy on evaluation data: 9597 / 10000
```

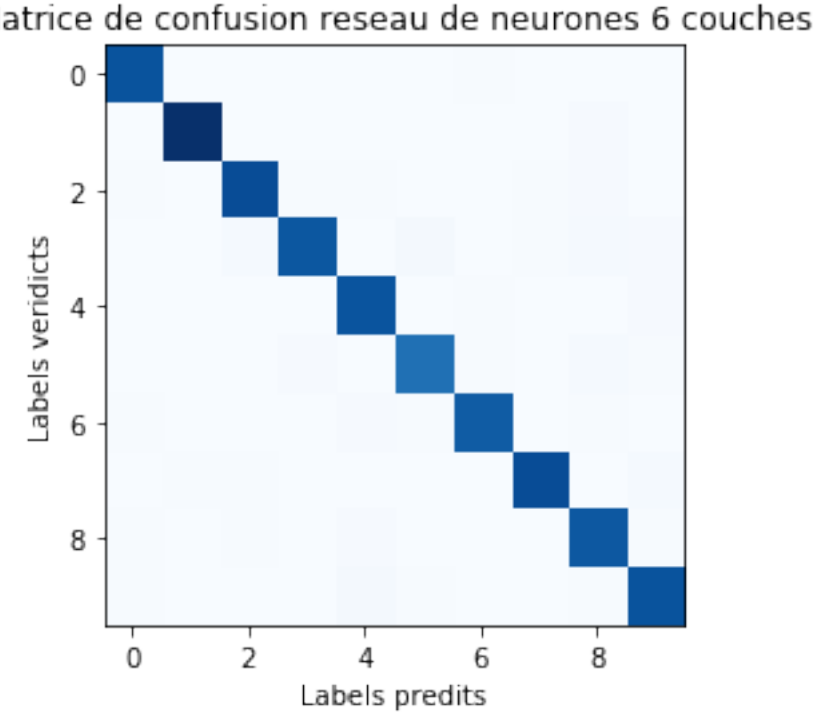
```
Epoch 23 training complete
Cost on training data: 0.297029228177
Accuracy on training data: 48871 / 50000
Cost on evaluation data: 0.973854976857
Accuracy on evaluation data: 9591 / 10000

Epoch 24 training complete
Cost on training data: 0.303118212355
Accuracy on training data: 48827 / 50000
Cost on evaluation data: 0.990605373687
Accuracy on evaluation data: 9575 / 10000

Epoch 25 training complete
Cost on training data: 0.277578888019
Accuracy on training data: 49071 / 50000
Cost on evaluation data: 0.965737926462
Accuracy on evaluation data: 9616 / 10000

Epoch 26 training complete
Cost on training data: 0.282520213849
Accuracy on training data: 49005 / 50000
Cost on evaluation data: 0.982081427201
Accuracy on evaluation data: 9588 / 10000

Epoch 27 training complete
Cost on training data: 0.280727553338
Accuracy on training data: 49024 / 50000
Cost on evaluation data: 0.9764133591
Accuracy on evaluation data: 9617 / 10000

Epoch 28 training complete
Cost on training data: 0.288841448128
Accuracy on training data: 48939 / 50000
Cost on evaluation data: 0.999105684977
Accuracy on evaluation data: 9569 / 10000

Epoch 29 training complete
Cost on training data: 0.278623815636
Accuracy on training data: 49023 / 50000
Cost on evaluation data: 0.970382322886
Accuracy on evaluation data: 9600 / 10000
```

[31]:
```
validation_data = list(validation_data)
test_data = list(test_data)
```

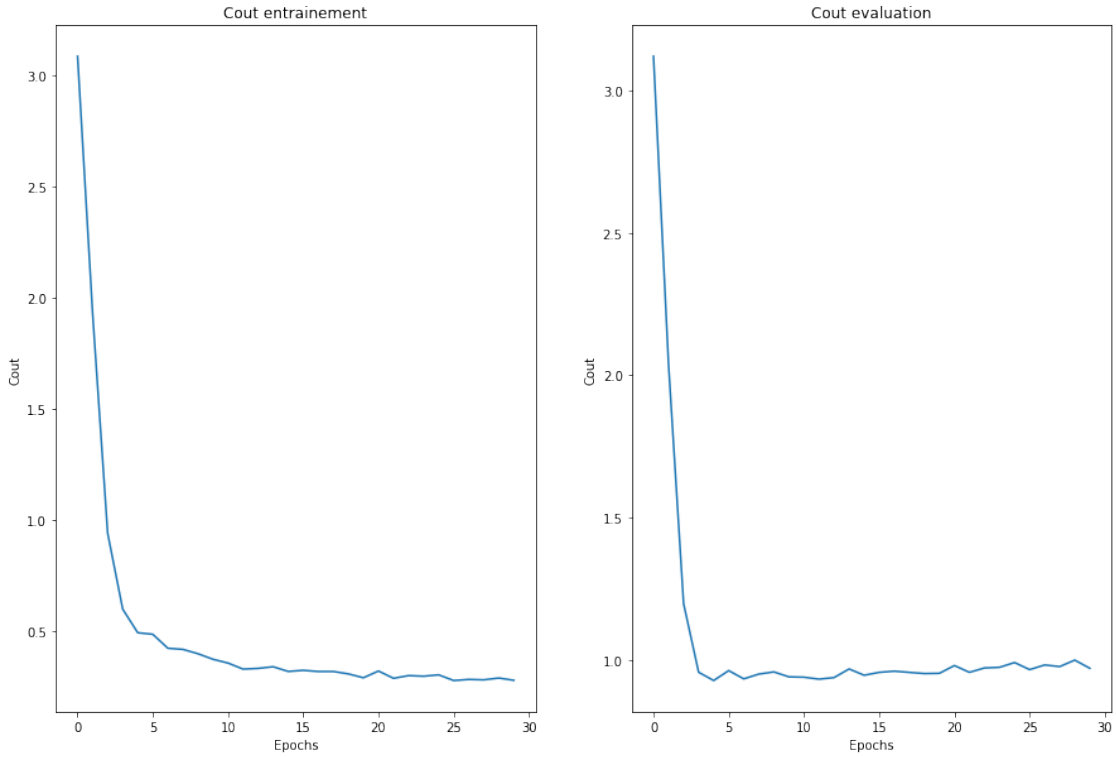
[32]:
```
plot_predict(test_data)
```



[33]:
```
plot_conf_matrix(test_data, plot_title="Matrice de confusion reseau de neurones␣
↪6 couches")
```

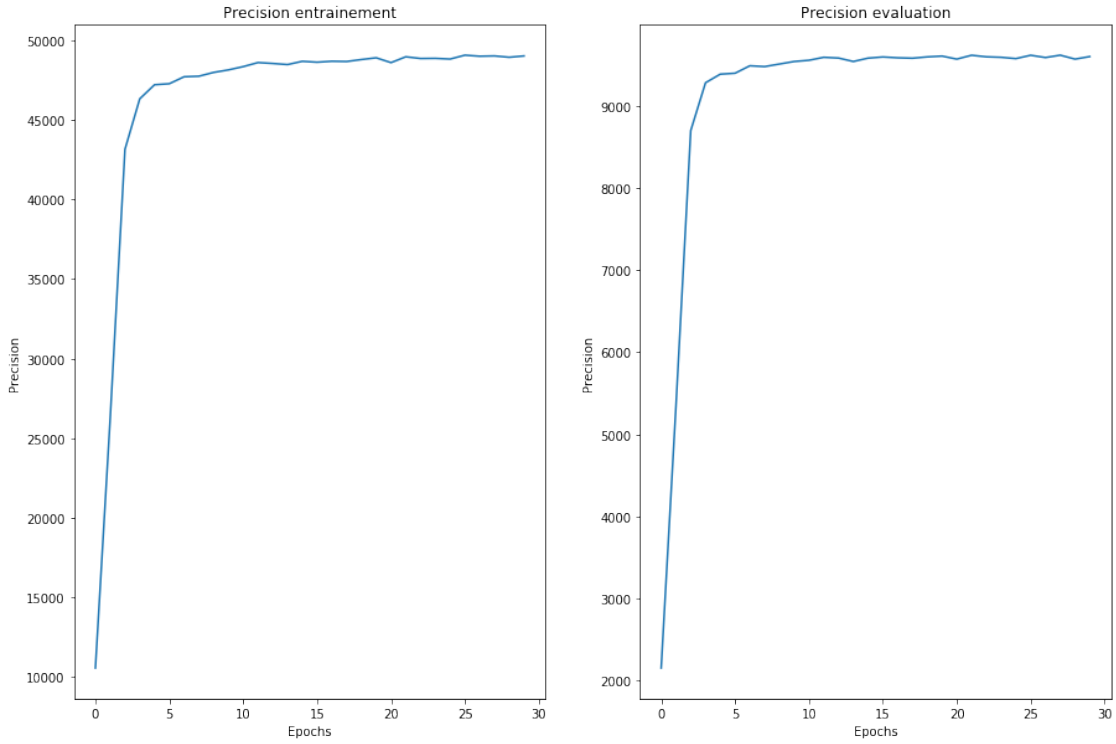


[34]:
```
subplots_data(
    [
        {
            "data": training_cost,
            "plot_tile": 'Cout entrainement',
            "label": ('Epochs', 'Cout'),
        },
        {
            "data": evaluation_cost,
            "plot_tile": 'Cout evaluation',
            "label": ('Epochs', 'Cout'),
        }
    ],
    1,
    2
)
```



[35]:
```
subplots_data(
    [
        {
            "data": training_accuracy,
```

```
            "plot_tile": 'Precision entrainement',
            "label": ('Epochs', 'Precision'),
        },
        {
            "data": evaluation_accuracy,
            "plot_tile": 'Precision evaluation',
            "label": ('Epochs', 'Precision'),
        }
    ],
    1,
    2
)
```



Réseau avec 12 couches

[36]:
```
import network2
import numpy as np
import mnist_loader
import matplotlib.pyplot as plt
from sklearn.metrics import confusion_matrix
```

[37]:
```
training_data, validation_data, test_data = mnist_loader.load_data_wrapper()
training_data = list(training_data)
```

[38]:
```
net, evaluation_cost, evaluation_accuracy, training_cost, training_accuracy =␣
↪train_network(
    training_data,
    validation_data,
    [784, 30, 30, 30, 30, 30, 30, 30, 30, 30, 30, 10]
)
```

```
Epoch 0 training complete
Cost on training data: 3.26798585338
Accuracy on training data: 4842 / 50000
Cost on evaluation data: 3.3271690653
Accuracy on evaluation data: 1009 / 10000

Epoch 1 training complete
Cost on training data: 3.26687868168
Accuracy on training data: 5101 / 50000
Cost on evaluation data: 3.32440803921
Accuracy on evaluation data: 1030 / 10000

Epoch 2 training complete
Cost on training data: 3.26783643083
Accuracy on training data: 5678 / 50000
Cost on evaluation data: 3.31969588127
Accuracy on evaluation data: 1064 / 10000

Epoch 3 training complete
Cost on training data: 3.26960444838
Accuracy on training data: 5175 / 50000
Cost on evaluation data: 3.31562832645
Accuracy on evaluation data: 1090 / 10000

Epoch 4 training complete
Cost on training data: 3.26293729959
Accuracy on training data: 5175 / 50000
Cost on evaluation data: 3.30657238814
Accuracy on evaluation data: 1090 / 10000

Epoch 5 training complete
Cost on training data: 3.26264571628
Accuracy on training data: 4859 / 50000
Cost on evaluation data: 3.30265941698
Accuracy on evaluation data: 983 / 10000

Epoch 6 training complete
Cost on training data: 3.2601026259
Accuracy on training data: 5678 / 50000
Cost on evaluation data: 3.29780103678
```

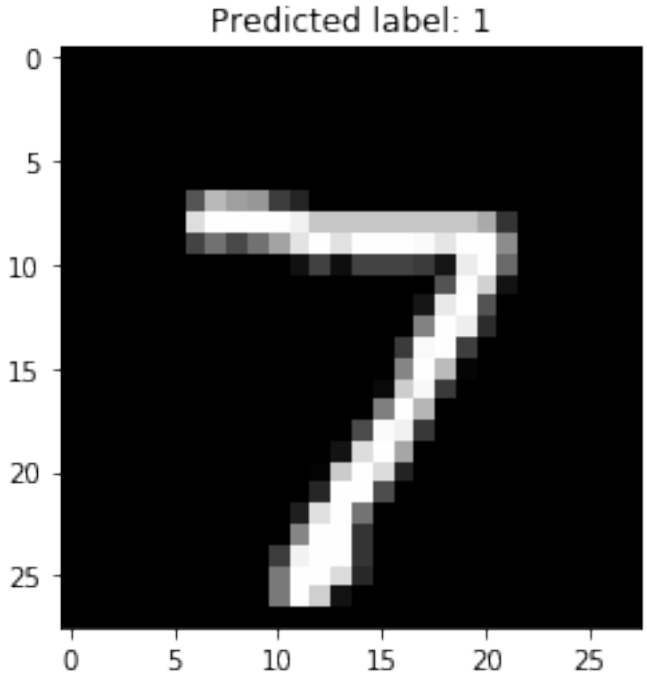
```
Accuracy on evaluation data: 1064 / 10000

Epoch 7 training complete
Cost on training data: 3.26106495859
Accuracy on training data: 5678 / 50000
Cost on evaluation data: 3.29618805513
Accuracy on evaluation data: 1064 / 10000

Epoch 8 training complete
Cost on training data: 3.26008275697
Accuracy on training data: 5678 / 50000
Cost on evaluation data: 3.29148731673
Accuracy on evaluation data: 1064 / 10000

Epoch 9 training complete
Cost on training data: 3.25806639988
Accuracy on training data: 4842 / 50000
Cost on evaluation data: 3.28640129563
Accuracy on evaluation data: 1009 / 10000

Epoch 10 training complete
Cost on training data: 3.25676950843
Accuracy on training data: 5678 / 50000
Cost on evaluation data: 3.28479419857
Accuracy on evaluation data: 1064 / 10000

Epoch 11 training complete
Cost on training data: 3.25765518529
Accuracy on training data: 5678 / 50000
Cost on evaluation data: 3.28282736473
Accuracy on evaluation data: 1064 / 10000

Epoch 12 training complete
Cost on training data: 3.2559986498
Accuracy on training data: 5678 / 50000
Cost on evaluation data: 3.2796110989
Accuracy on evaluation data: 1064 / 10000

Epoch 13 training complete
Cost on training data: 3.25537464566
Accuracy on training data: 5678 / 50000
Cost on evaluation data: 3.27780944371
Accuracy on evaluation data: 1064 / 10000

Epoch 14 training complete
Cost on training data: 3.25677379497
Accuracy on training data: 5175 / 50000
Cost on evaluation data: 3.275325729
```

```
Accuracy on evaluation data: 1090 / 10000

Epoch 15 training complete
Cost on training data: 3.25429080207
Accuracy on training data: 5678 / 50000
Cost on evaluation data: 3.27252615566
Accuracy on evaluation data: 1064 / 10000

Epoch 16 training complete
Cost on training data: 3.25404382846
Accuracy on training data: 5678 / 50000
Cost on evaluation data: 3.27166915038
Accuracy on evaluation data: 1064 / 10000

Epoch 17 training complete
Cost on training data: 3.25672580263
Accuracy on training data: 5678 / 50000
Cost on evaluation data: 3.27538727204
Accuracy on evaluation data: 1064 / 10000

Epoch 18 training complete
Cost on training data: 3.25385857625
Accuracy on training data: 5678 / 50000
Cost on evaluation data: 3.2686712063
Accuracy on evaluation data: 1064 / 10000

Epoch 19 training complete
Cost on training data: 3.25340378821
Accuracy on training data: 5678 / 50000
Cost on evaluation data: 3.26669547903
Accuracy on evaluation data: 1064 / 10000

Epoch 20 training complete
Cost on training data: 3.25444706001
Accuracy on training data: 4932 / 50000
Cost on evaluation data: 3.26700419182
Accuracy on evaluation data: 991 / 10000

Epoch 21 training complete
Cost on training data: 3.25508555634
Accuracy on training data: 4932 / 50000
Cost on evaluation data: 3.26812039578
Accuracy on evaluation data: 991 / 10000

Epoch 22 training complete
Cost on training data: 3.25366721077
Accuracy on training data: 5101 / 50000
Cost on evaluation data: 3.26429074497
```

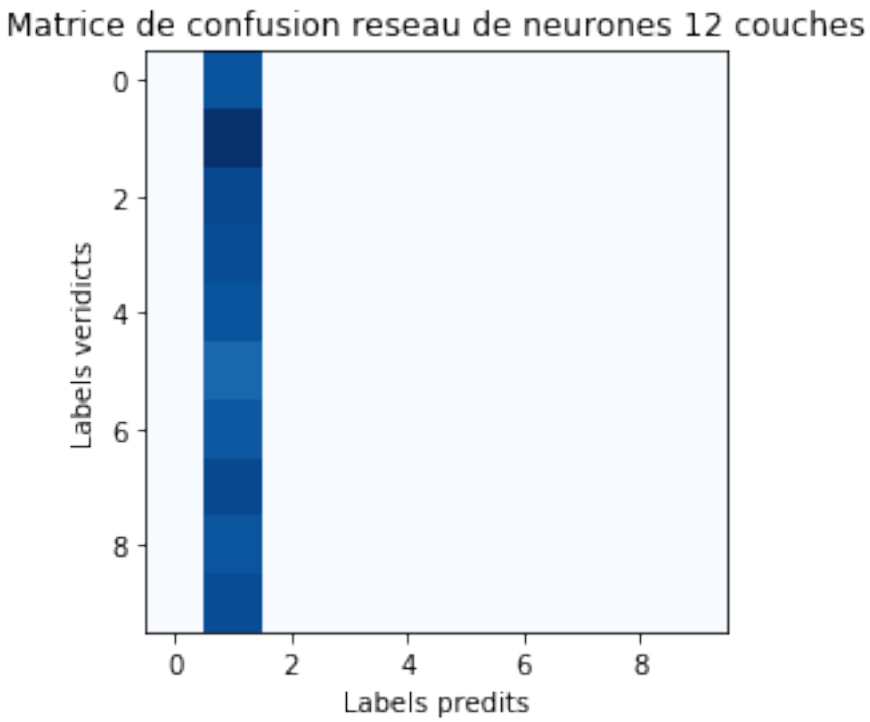
```
Accuracy on evaluation data: 1030 / 10000

Epoch 23 training complete
Cost on training data: 3.25192926704
Accuracy on training data: 5678 / 50000
Cost on evaluation data: 3.26259213795
Accuracy on evaluation data: 1064 / 10000

Epoch 24 training complete
Cost on training data: 3.25185694184
Accuracy on training data: 5678 / 50000
Cost on evaluation data: 3.26258290467
Accuracy on evaluation data: 1064 / 10000

Epoch 25 training complete
Cost on training data: 3.25427106282
Accuracy on training data: 5101 / 50000
Cost on evaluation data: 3.26274404388
Accuracy on evaluation data: 1030 / 10000

Epoch 26 training complete
Cost on training data: 3.25414189235
Accuracy on training data: 5101 / 50000
Cost on evaluation data: 3.2633135563
Accuracy on evaluation data: 1030 / 10000

Epoch 27 training complete
Cost on training data: 3.25325682925
Accuracy on training data: 5101 / 50000
Cost on evaluation data: 3.26087673623
Accuracy on evaluation data: 1030 / 10000

Epoch 28 training complete
Cost on training data: 3.25310643516
Accuracy on training data: 5678 / 50000
Cost on evaluation data: 3.26207793544
Accuracy on evaluation data: 1064 / 10000

Epoch 29 training complete
Cost on training data: 3.25408510987
Accuracy on training data: 5678 / 50000
Cost on evaluation data: 3.26227596015
Accuracy on evaluation data: 1064 / 10000
```

[39]:
```
validation_data = list(validation_data)
test_data = list(test_data)
```

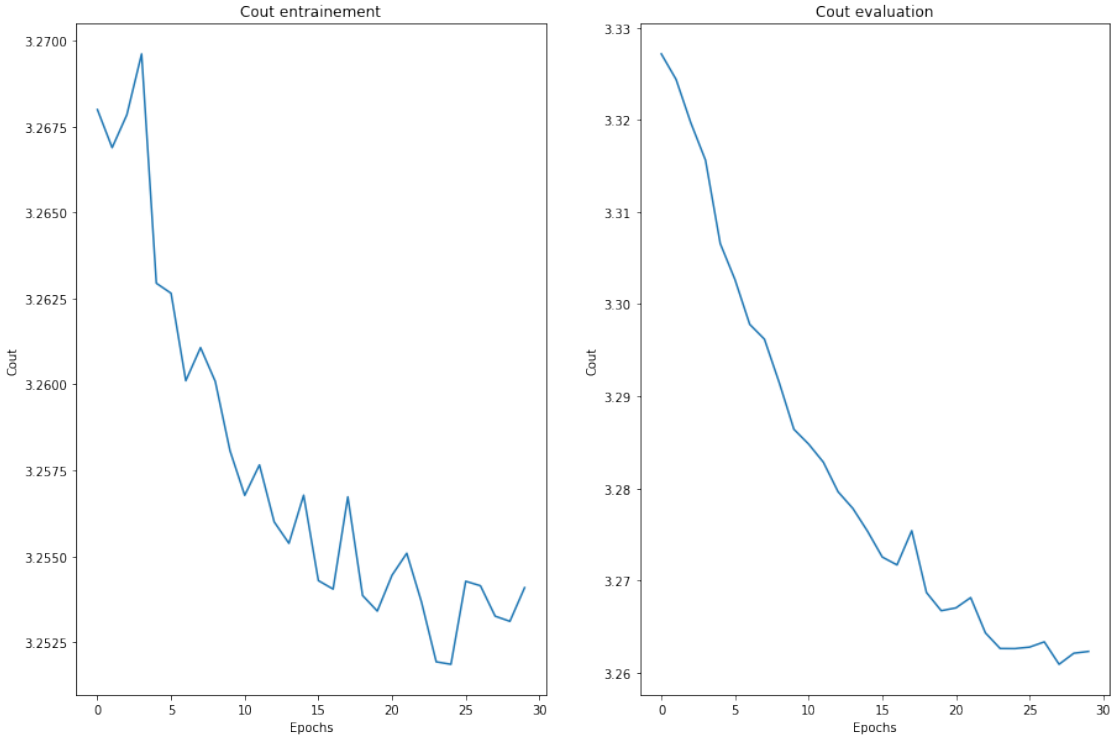
[40]:
```
plot_predict(test_data)
```



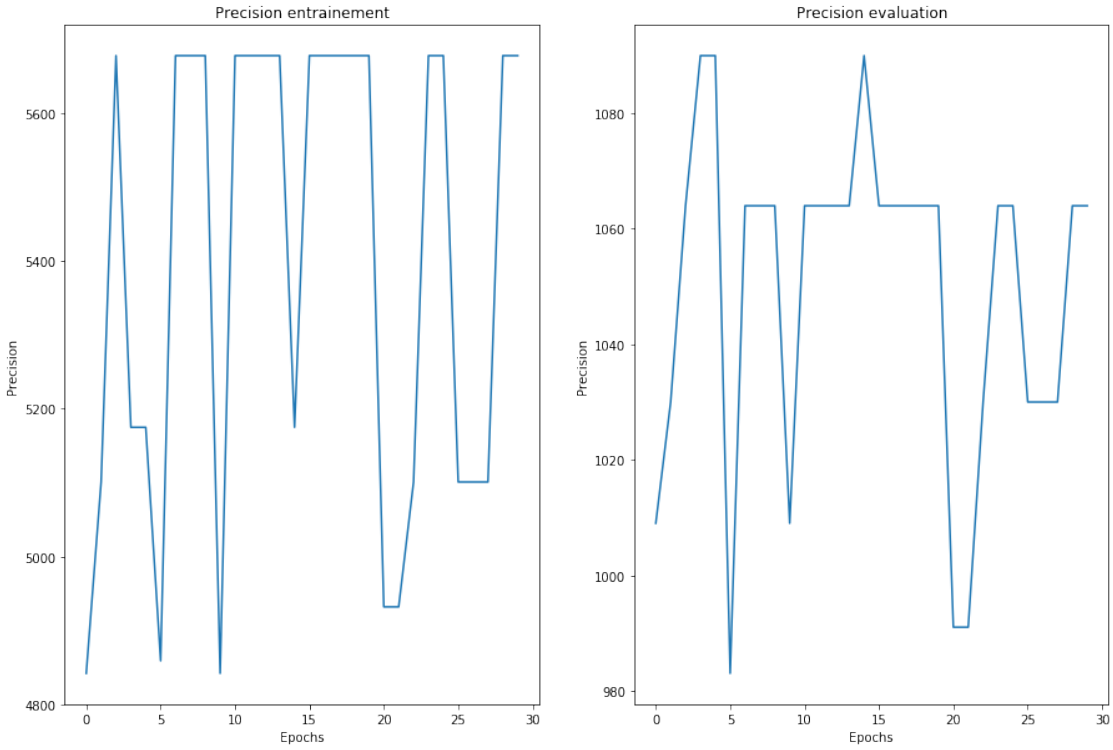
[41]:
```
plot_conf_matrix(test_data, plot_title="Matrice de confusion reseau de neurones␣
↪12 couches")
```



[42]:
```
subplots_data(
    [
        {
            "data": training_cost,
            "plot_tile": 'Cout entrainement',
            "label": ('Epochs', 'Cout'),
        },
        {
            "data": evaluation_cost,
            "plot_tile": 'Cout evaluation',
            "label": ('Epochs', 'Cout'),
        }
    ],
    1,
    2
)
```



[43]:
```
subplots_data(
    [
        {
            "data": training_accuracy,
            "plot_tile": 'Precision entrainement',
            "label": ('Epochs', 'Precision'),
        },
        {
            "data": evaluation_accuracy,
            "plot_tile": 'Precision evaluation',
            "label": ('Epochs', 'Precision'),
        }
    ],
    1,
    2
)
```



2.2.4 Réseaux de neurones avec couches de convolution

[44]:
```
import network3
from network3 import ConvPoolLayer, FullyConnectedLayer, SoftmaxLayer, ReLU
import numpy as np
import matplotlib.pyplot as plt
```

```
Trying to run under a GPU.  If this is not desired, then modify network3.py
to set the GPU flag to False.
```

[45]:
```
training_data, validation_data, test_data = network3.load_data_shared()
```

[46]:
```
epochs = 30
mini_batch_size = 10
eta = 0.03
```

[47]:
```
net = network3.Network([
    ConvPoolLayer(image_shape=(mini_batch_size, 1, 28, 28),
                  filter_shape=(20, 1, 5, 5),
                  poolsize=(2, 2),
                  activation_fn=ReLU),
    ConvPoolLayer(image_shape=(mini_batch_size, 20, 12, 12),
```

```
                  filter_shape=(40, 20, 5, 5),
                  poolsize=(2, 2),
                  activation_fn=ReLU),
    FullyConnectedLayer(n_in=40*4*4, n_out=100, activation_fn=ReLU),
    SoftmaxLayer(n_in=100, n_out=10)], mini_batch_size)
validation_accuracy, test_accuracy = net.SGD(training_data, epochs,␣
 ↪mini_batch_size, eta, validation_data, test_data, lmbda=0.1)
```

```
Training mini-batch number 0
Training mini-batch number 1000
Training mini-batch number 2000
Training mini-batch number 3000
Training mini-batch number 4000
Epoch 0: validation accuracy 97.85%
This is the best validation accuracy to date.
The corresponding test accuracy is 97.43%
Training mini-batch number 5000
Training mini-batch number 6000
Training mini-batch number 7000
Training mini-batch number 8000
Training mini-batch number 9000
Epoch 1: validation accuracy 98.27%
This is the best validation accuracy to date.
The corresponding test accuracy is 98.14%
Training mini-batch number 10000
Training mini-batch number 11000
Training mini-batch number 12000
Training mini-batch number 13000
Training mini-batch number 14000
Epoch 2: validation accuracy 98.40%
This is the best validation accuracy to date.
The corresponding test accuracy is 98.38%
Training mini-batch number 15000
Training mini-batch number 16000
Training mini-batch number 17000
Training mini-batch number 18000
Training mini-batch number 19000
Epoch 3: validation accuracy 98.58%
This is the best validation accuracy to date.
The corresponding test accuracy is 98.61%
Training mini-batch number 20000
Training mini-batch number 21000
Training mini-batch number 22000
Training mini-batch number 23000
Training mini-batch number 24000
Epoch 4: validation accuracy 98.62%
This is the best validation accuracy to date.
```

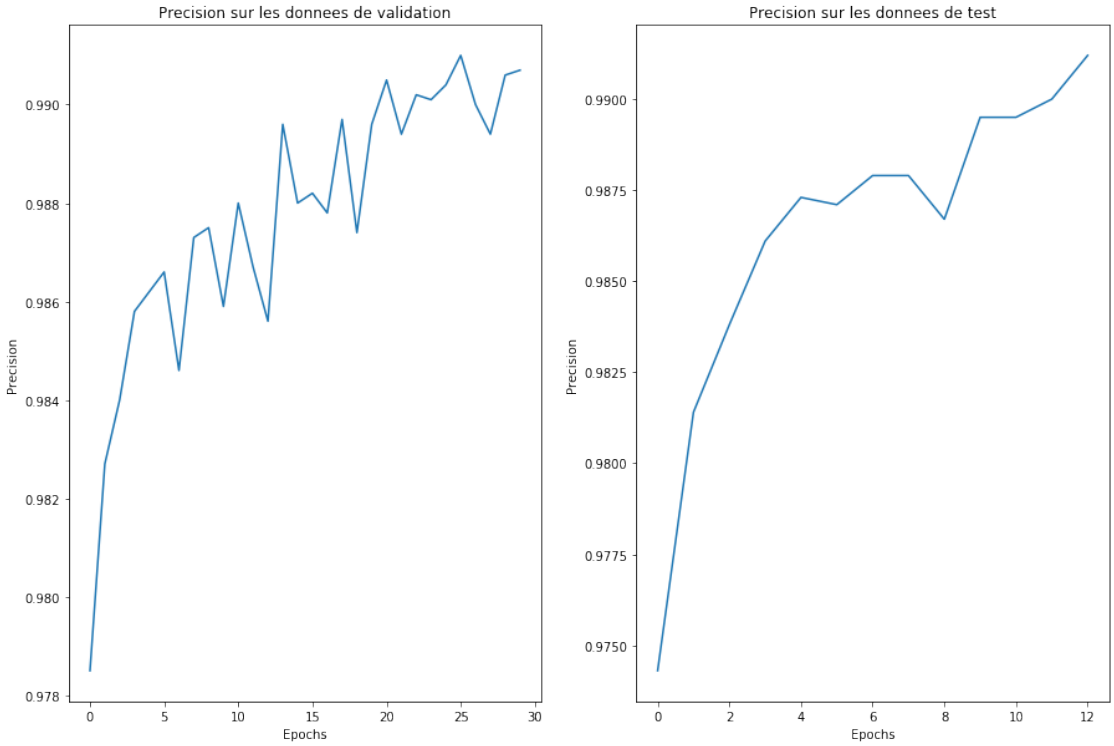
```
The corresponding test accuracy is 98.73%
Training mini-batch number 25000
Training mini-batch number 26000
Training mini-batch number 27000
Training mini-batch number 28000
Training mini-batch number 29000
Epoch 5: validation accuracy 98.66%
This is the best validation accuracy to date.
The corresponding test accuracy is 98.71%
Training mini-batch number 30000
Training mini-batch number 31000
Training mini-batch number 32000
Training mini-batch number 33000
Training mini-batch number 34000
Epoch 6: validation accuracy 98.46%
Training mini-batch number 35000
Training mini-batch number 36000
Training mini-batch number 37000
Training mini-batch number 38000
Training mini-batch number 39000
Epoch 7: validation accuracy 98.73%
This is the best validation accuracy to date.
The corresponding test accuracy is 98.79%
Training mini-batch number 40000
Training mini-batch number 41000
Training mini-batch number 42000
Training mini-batch number 43000
Training mini-batch number 44000
Epoch 8: validation accuracy 98.75%
This is the best validation accuracy to date.
The corresponding test accuracy is 98.79%
Training mini-batch number 45000
Training mini-batch number 46000
Training mini-batch number 47000
Training mini-batch number 48000
Training mini-batch number 49000
Epoch 9: validation accuracy 98.59%
Training mini-batch number 50000
Training mini-batch number 51000
Training mini-batch number 52000
Training mini-batch number 53000
Training mini-batch number 54000
Epoch 10: validation accuracy 98.80%
This is the best validation accuracy to date.
The corresponding test accuracy is 98.67%
Training mini-batch number 55000
Training mini-batch number 56000
Training mini-batch number 57000
```

```
Training mini-batch number 58000
Training mini-batch number 59000
Epoch 11: validation accuracy 98.67%
Training mini-batch number 60000
Training mini-batch number 61000
Training mini-batch number 62000
Training mini-batch number 63000
Training mini-batch number 64000
Epoch 12: validation accuracy 98.56%
Training mini-batch number 65000
Training mini-batch number 66000
Training mini-batch number 67000
Training mini-batch number 68000
Training mini-batch number 69000
Epoch 13: validation accuracy 98.96%
This is the best validation accuracy to date.
The corresponding test accuracy is 98.95%
Training mini-batch number 70000
Training mini-batch number 71000
Training mini-batch number 72000
Training mini-batch number 73000
Training mini-batch number 74000
Epoch 14: validation accuracy 98.80%
Training mini-batch number 75000
Training mini-batch number 76000
Training mini-batch number 77000
Training mini-batch number 78000
Training mini-batch number 79000
Epoch 15: validation accuracy 98.82%
Training mini-batch number 80000
Training mini-batch number 81000
Training mini-batch number 82000
Training mini-batch number 83000
Training mini-batch number 84000
Epoch 16: validation accuracy 98.78%
Training mini-batch number 85000
Training mini-batch number 86000
Training mini-batch number 87000
Training mini-batch number 88000
Training mini-batch number 89000
Epoch 17: validation accuracy 98.97%
This is the best validation accuracy to date.
The corresponding test accuracy is 98.95%
Training mini-batch number 90000
Training mini-batch number 91000
Training mini-batch number 92000
Training mini-batch number 93000
Training mini-batch number 94000
```

```
Epoch 18: validation accuracy 98.74%
Training mini-batch number 95000
Training mini-batch number 96000
Training mini-batch number 97000
Training mini-batch number 98000
Training mini-batch number 99000
Epoch 19: validation accuracy 98.96%
Training mini-batch number 100000
Training mini-batch number 101000
Training mini-batch number 102000
Training mini-batch number 103000
Training mini-batch number 104000
Epoch 20: validation accuracy 99.05%
This is the best validation accuracy to date.
The corresponding test accuracy is 99.00%
Training mini-batch number 105000
Training mini-batch number 106000
Training mini-batch number 107000
Training mini-batch number 108000
Training mini-batch number 109000
Epoch 21: validation accuracy 98.94%
Training mini-batch number 110000
Training mini-batch number 111000
Training mini-batch number 112000
Training mini-batch number 113000
Training mini-batch number 114000
Epoch 22: validation accuracy 99.02%
Training mini-batch number 115000
Training mini-batch number 116000
Training mini-batch number 117000
Training mini-batch number 118000
Training mini-batch number 119000
Epoch 23: validation accuracy 99.01%
Training mini-batch number 120000
Training mini-batch number 121000
Training mini-batch number 122000
Training mini-batch number 123000
Training mini-batch number 124000
Epoch 24: validation accuracy 99.04%
Training mini-batch number 125000
Training mini-batch number 126000
Training mini-batch number 127000
Training mini-batch number 128000
Training mini-batch number 129000
Epoch 25: validation accuracy 99.10%
This is the best validation accuracy to date.
The corresponding test accuracy is 99.12%
Training mini-batch number 130000
```

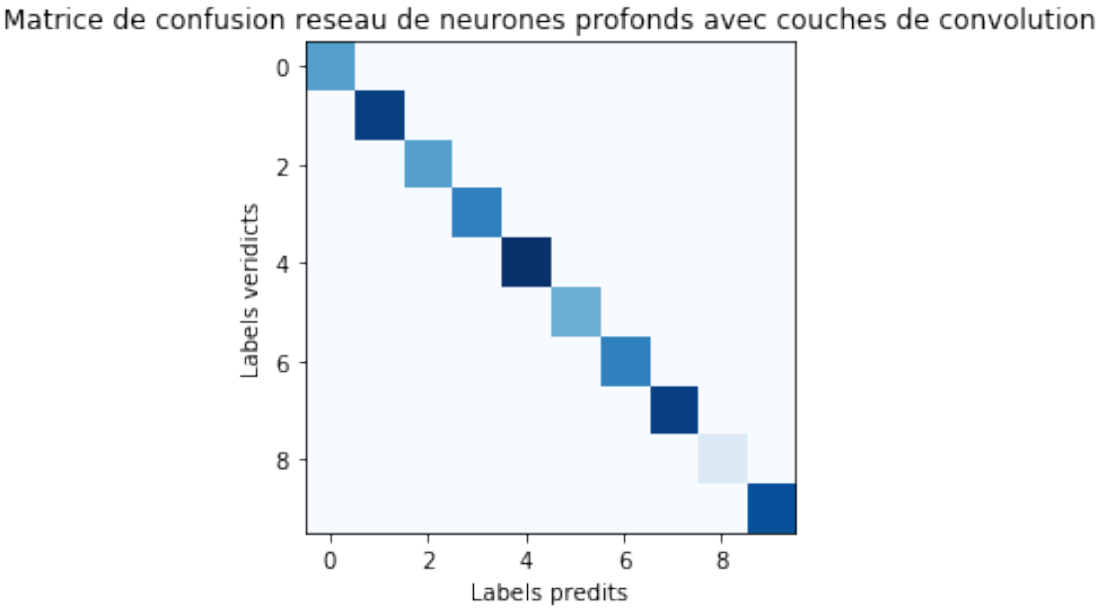
```
Training mini-batch number 131000
Training mini-batch number 132000
Training mini-batch number 133000
Training mini-batch number 134000
Epoch 26: validation accuracy 99.00%
Training mini-batch number 135000
Training mini-batch number 136000
Training mini-batch number 137000
Training mini-batch number 138000
Training mini-batch number 139000
Epoch 27: validation accuracy 98.94%
Training mini-batch number 140000
Training mini-batch number 141000
Training mini-batch number 142000
Training mini-batch number 143000
Training mini-batch number 144000
Epoch 28: validation accuracy 99.06%
Training mini-batch number 145000
Training mini-batch number 146000
Training mini-batch number 147000
Training mini-batch number 148000
Training mini-batch number 149000
Epoch 29: validation accuracy 99.07%
Finished training network.
Best validation accuracy of 99.10% obtained at iteration 129999
Corresponding test accuracy of 99.12%
```

[48]:
```
subplots_data(
    [
        {
            "data": validation_accuracy,
            "plot_tile": 'Precision sur les donnees de validation',
            "label": ('Epochs', 'Precision'),
        },
        {
            "data": test_accuracy,
            "plot_tile": 'Precision sur les donnees de test',
            "label": ('Epochs', 'Precision'),
        }
    ],
    1,
    2
)
```



[49]:
```
test_x, test_y = test_data
test_y = test_y.eval()
predictions = np.concatenate([net.test_mb_predictions(i) for i in range (0,␣
↪mini_batch_size + 1)])
```

[50]:
```
conf_mat = confusion_matrix(test_y[0:len(predictions)], predictions)
plt.imshow(conf_mat, cmap=plt.cm.Blues)
plt.xlabel("Labels predits")
plt.ylabel("Labels veridicts")
plt.title("Matrice de confusion reseau de neurones profonds avec couches de␣
↪convolution")
plt.show()
```


Matrice de confusion reseau de neurones profonds avec couches de convolution
Labels veridicts
Labels predits
0
2
4
6
8
0
2
4
6
8