DaSilva_Bailleul_Classification_Nombres_Rapport_DeepLearning_E4FI

April 5, 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 20ème 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 20ème epoch. Cela suggère que le modèle est capable de généraliser à 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 semple 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 perceptron multicouche, les résultats obtenus avec les couches de convolution sont bien meilleurs. En utilisant un simple perceptron multicouche, 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):
    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("Matrice de confusion Perceptron \"de base\"")
    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

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

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

```
[140]: 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: 0.746188850334
Accuracy on training data: 45034 / 50000
Cost on evaluation data: 0.744207926242
Accuracy on evaluation data: 9101 / 10000

Epoch 1 training complete
Cost on training data: 0.706881397008
Accuracy on training data: 45666 / 50000
```

```
Cost on evaluation data: 0.724052536848
Accuracy on evaluation data: 9168 / 10000

Epoch 2 training complete
Cost on training data: 0.724234533698
Accuracy on training data: 45195 / 50000
Cost on evaluation data: 0.753111989988
Accuracy on evaluation data: 9089 / 10000

Epoch 3 training complete
Cost on training data: 0.683776944634
Accuracy on training data: 45702 / 50000
Cost on evaluation data: 0.731613197544
Accuracy on evaluation data: 9180 / 10000

Epoch 4 training complete
Cost on training data: 0.677054027373
Accuracy on training data: 45727 / 50000
Cost on evaluation data: 0.733323108012
Accuracy on evaluation data: 9171 / 10000

Epoch 5 training complete
Cost on training data: 0.670701679734
Accuracy on training data: 45814 / 50000
Cost on evaluation data: 0.729219607948
Accuracy on evaluation data: 9193 / 10000

Epoch 6 training complete
Cost on training data: 0.682419082598
Accuracy on training data: 45780 / 50000
Cost on evaluation data: 0.749833805847
Accuracy on evaluation data: 9190 / 10000

Epoch 7 training complete
Cost on training data: 0.677371127142
Accuracy on training data: 45754 / 50000
Cost on evaluation data: 0.750875176989
Accuracy on evaluation data: 9192 / 10000

Epoch 8 training complete
Cost on training data: 0.675005656837
Accuracy on training data: 45856 / 50000
Cost on evaluation data: 0.75477876234
Accuracy on evaluation data: 9202 / 10000

Epoch 9 training complete
Cost on training data: 0.676051666085
Accuracy on training data: 45840 / 50000
```

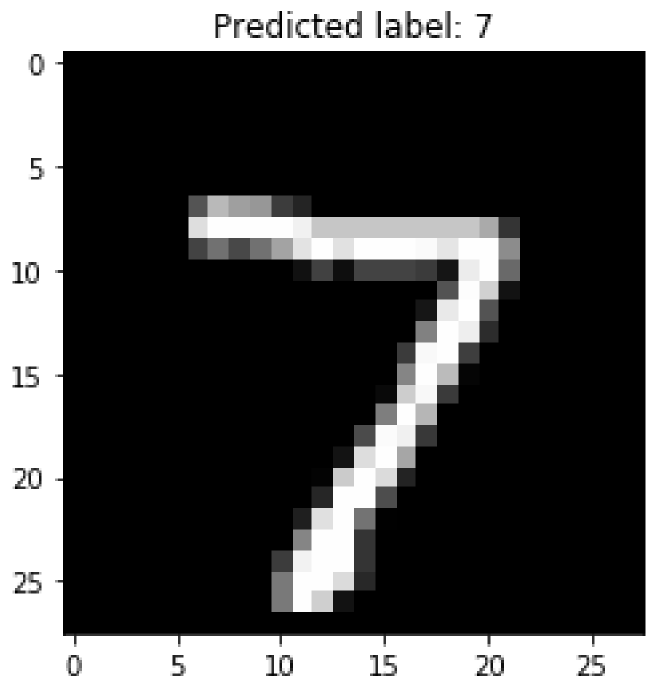
```
Cost on evaluation data: 0.762277266857
Accuracy on evaluation data: 9180 / 10000

Epoch 10 training complete
Cost on training data: 0.742671071106
Accuracy on training data: 45381 / 50000
Cost on evaluation data: 0.824286935205
Accuracy on evaluation data: 9116 / 10000

Epoch 11 training complete
Cost on training data: 0.695208141693
Accuracy on training data: 45722 / 50000
Cost on evaluation data: 0.790773232631
Accuracy on evaluation data: 9184 / 10000

Epoch 12 training complete
Cost on training data: 0.691588622204
Accuracy on training data: 45720 / 50000
Cost on evaluation data: 0.77656226187
Accuracy on evaluation data: 9159 / 10000

Epoch 13 training complete
Cost on training data: 0.691284518669
Accuracy on training data: 45782 / 50000
Cost on evaluation data: 0.77880136363
Accuracy on evaluation data: 9191 / 10000

Epoch 14 training complete
Cost on training data: 0.696192673026
Accuracy on training data: 45773 / 50000
Cost on evaluation data: 0.80174608486
Accuracy on evaluation data: 9167 / 10000

Epoch 15 training complete
Cost on training data: 0.673587216872
Accuracy on training data: 45964 / 50000
Cost on evaluation data: 0.770443183278
Accuracy on evaluation data: 9213 / 10000

Epoch 16 training complete
Cost on training data: 0.693389039756
Accuracy on training data: 45711 / 50000
Cost on evaluation data: 0.809113737574
Accuracy on evaluation data: 9177 / 10000

Epoch 17 training complete
Cost on training data: 0.69282207146
Accuracy on training data: 45608 / 50000
```

```
Cost on evaluation data: 0.808436859649
Accuracy on evaluation data: 9137 / 10000

Epoch 18 training complete
Cost on training data: 0.678526604587
Accuracy on training data: 45929 / 50000
Cost on evaluation data: 0.792784925534
Accuracy on evaluation data: 9204 / 10000

Epoch 19 training complete
Cost on training data: 0.67511433091
Accuracy on training data: 45944 / 50000
Cost on evaluation data: 0.790260102428
Accuracy on evaluation data: 9197 / 10000

Epoch 20 training complete
Cost on training data: 0.689750053812
Accuracy on training data: 45770 / 50000
Cost on evaluation data: 0.812366357066
Accuracy on evaluation data: 9157 / 10000

Epoch 21 training complete
Cost on training data: 0.670385869314
Accuracy on training data: 45920 / 50000
Cost on evaluation data: 0.794466680027
Accuracy on evaluation data: 9185 / 10000

Epoch 22 training complete
Cost on training data: 0.687635861951
Accuracy on training data: 45768 / 50000
Cost on evaluation data: 0.807256987427
Accuracy on evaluation data: 9167 / 10000

Epoch 23 training complete
Cost on training data: 0.667073359073
Accuracy on training data: 45965 / 50000
Cost on evaluation data: 0.791026274324
Accuracy on evaluation data: 9203 / 10000

Epoch 24 training complete
Cost on training data: 0.668984141412
Accuracy on training data: 46043 / 50000
Cost on evaluation data: 0.787371390429
Accuracy on evaluation data: 9211 / 10000

Epoch 25 training complete
Cost on training data: 0.678196405765
Accuracy on training data: 45882 / 50000
```

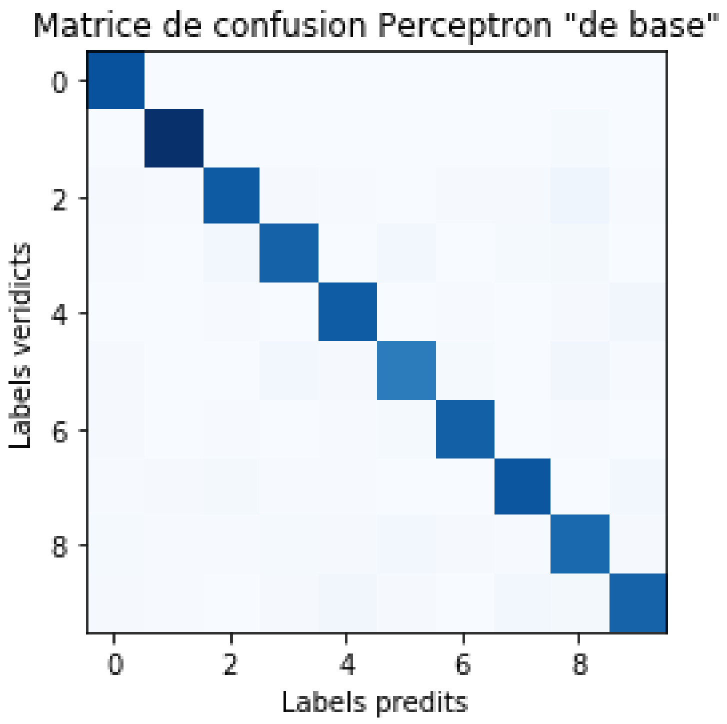
```
Cost on evaluation data: 0.798400361817
Accuracy on evaluation data: 9199 / 10000

Epoch 26 training complete
Cost on training data: 0.674086354197
Accuracy on training data: 45909 / 50000
Cost on evaluation data: 0.798993801184
Accuracy on evaluation data: 9189 / 10000

Epoch 27 training complete
Cost on training data: 0.668985390131
Accuracy on training data: 45967 / 50000
Cost on evaluation data: 0.794566655495
Accuracy on evaluation data: 9188 / 10000

Epoch 28 training complete
Cost on training data: 0.683149245942
Accuracy on training data: 45860 / 50000
Cost on evaluation data: 0.80421369565
Accuracy on evaluation data: 9175 / 10000

Epoch 29 training complete
Cost on training data: 0.671487285631
Accuracy on training data: 46001 / 50000
Cost on evaluation data: 0.79645188695
Accuracy on evaluation data: 9219 / 10000
```

[141]:
```
validation_data = list(validation_data)
test_data = list(test_data)
```

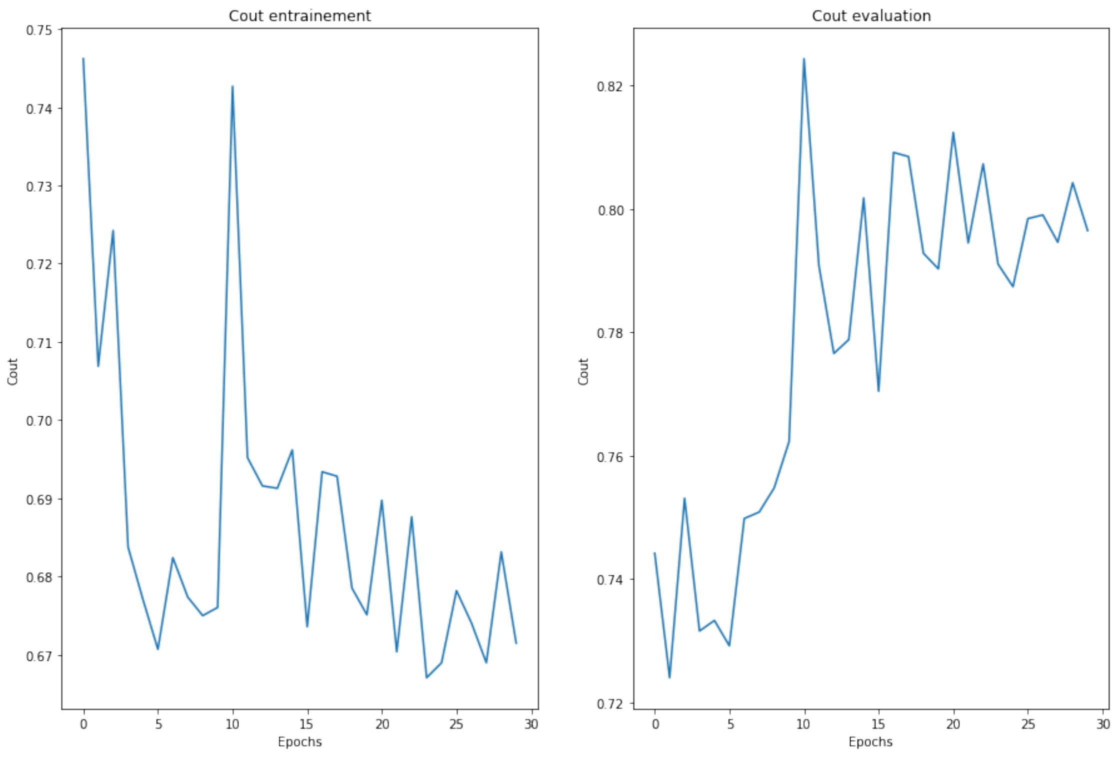
[142]:
```
plot_predict(test_data)
```



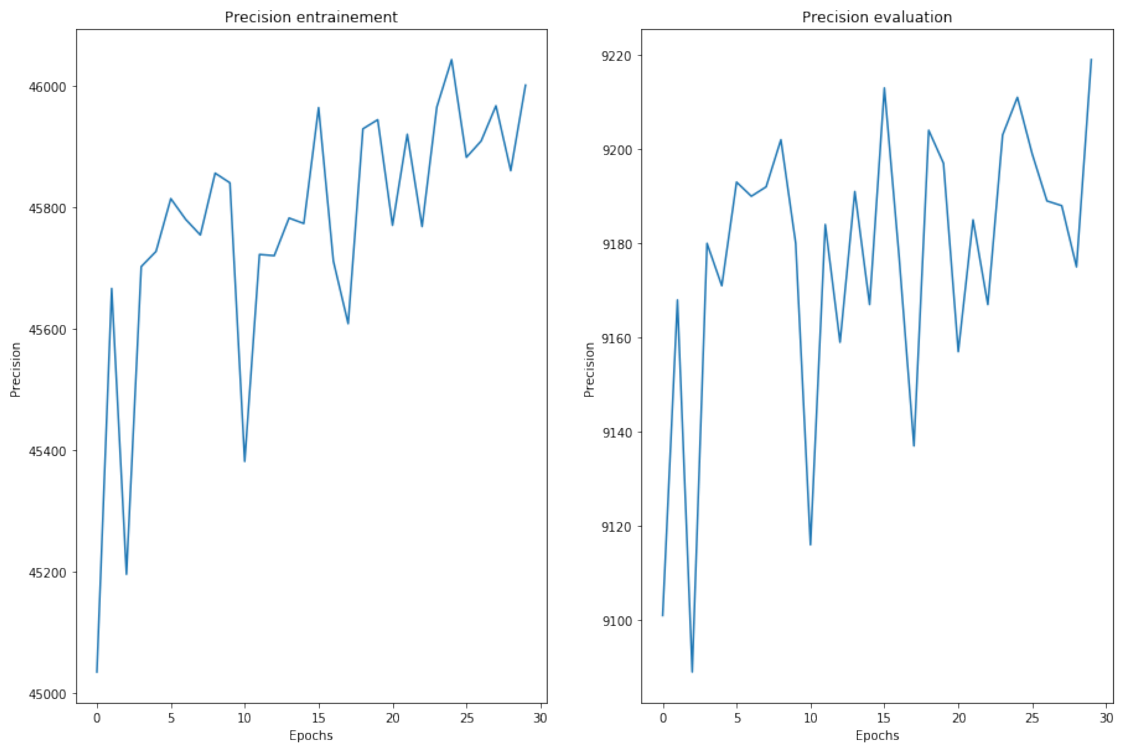
```
[143]: plot_conf_matrix(test_data)
```



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



```
[145]: 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: 0.563062915394
Accuracy on training data: 46151 / 50000
Cost on evaluation data: 0.656524027277
Accuracy on evaluation data: 9295 / 10000

Epoch 1 training complete
Cost on training data: 0.488914555612
Accuracy on training data: 46728 / 50000
Cost on evaluation data: 0.650271941568
Accuracy on evaluation data: 9374 / 10000

Epoch 2 training complete
Cost on training data: 0.430848814927
Accuracy on training data: 47328 / 50000
Cost on evaluation data: 0.641522082645
Accuracy on evaluation data: 9478 / 10000

Epoch 3 training complete
Cost on training data: 0.397032395262
Accuracy on training data: 47593 / 50000
Cost on evaluation data: 0.651342874684
Accuracy on evaluation data: 9504 / 10000

Epoch 4 training complete
Cost on training data: 0.383679047258
Accuracy on training data: 47819 / 50000
Cost on evaluation data: 0.663734574577
Accuracy on evaluation data: 9540 / 10000

Epoch 5 training complete
Cost on training data: 0.365473494256
Accuracy on training data: 47935 / 50000
Cost on evaluation data: 0.672476322297
Accuracy on evaluation data: 9555 / 10000

Epoch 6 training complete
```

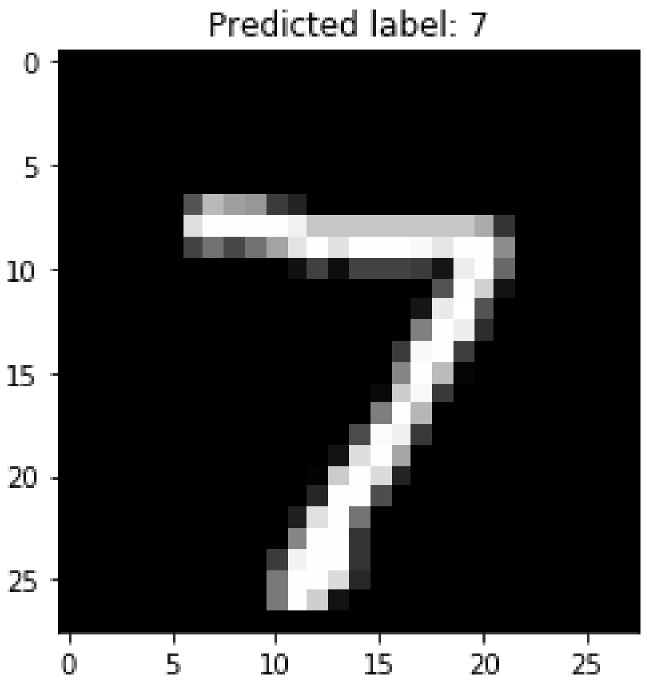
```
Cost on training data: 0.356629278193
Accuracy on training data: 48077 / 50000
Cost on evaluation data: 0.685062910294
Accuracy on evaluation data: 9563 / 10000

Epoch 7 training complete
Cost on training data: 0.364255253018
Accuracy on training data: 48070 / 50000
Cost on evaluation data: 0.711229844262
Accuracy on evaluation data: 9564 / 10000

Epoch 8 training complete
Cost on training data: 0.34429501065
Accuracy on training data: 48256 / 50000
Cost on evaluation data: 0.709168863386
Accuracy on evaluation data: 9585 / 10000

Epoch 9 training complete
Cost on training data: 0.342557611402
Accuracy on training data: 48271 / 50000
Cost on evaluation data: 0.718688522659
Accuracy on evaluation data: 9588 / 10000

Epoch 10 training complete
Cost on training data: 0.338733211633
Accuracy on training data: 48383 / 50000
Cost on evaluation data: 0.731569759464
Accuracy on evaluation data: 9586 / 10000

Epoch 11 training complete
Cost on training data: 0.329927627096
Accuracy on training data: 48418 / 50000
Cost on evaluation data: 0.734749306568
Accuracy on evaluation data: 9588 / 10000

Epoch 12 training complete
Cost on training data: 0.32567003085
Accuracy on training data: 48477 / 50000
Cost on evaluation data: 0.739494463645
Accuracy on evaluation data: 9608 / 10000

Epoch 13 training complete
Cost on training data: 0.332638022316
Accuracy on training data: 48449 / 50000
Cost on evaluation data: 0.753157768447
Accuracy on evaluation data: 9613 / 10000

Epoch 14 training complete
```

```
Cost on training data: 0.327778811802
Accuracy on training data: 48495 / 50000
Cost on evaluation data: 0.75645517844
Accuracy on evaluation data: 9617 / 10000

Epoch 15 training complete
Cost on training data: 0.321911702998
Accuracy on training data: 48604 / 50000
Cost on evaluation data: 0.757983311484
Accuracy on evaluation data: 9622 / 10000

Epoch 16 training complete
Cost on training data: 0.323539468845
Accuracy on training data: 48544 / 50000
Cost on evaluation data: 0.768434181159
Accuracy on evaluation data: 9609 / 10000

Epoch 17 training complete
Cost on training data: 0.322589419626
Accuracy on training data: 48600 / 50000
Cost on evaluation data: 0.771002576603
Accuracy on evaluation data: 9611 / 10000

Epoch 18 training complete
Cost on training data: 0.316387371657
Accuracy on training data: 48654 / 50000
Cost on evaluation data: 0.773977271089
Accuracy on evaluation data: 9632 / 10000

Epoch 19 training complete
Cost on training data: 0.312161733157
Accuracy on training data: 48696 / 50000
Cost on evaluation data: 0.777857086156
Accuracy on evaluation data: 9613 / 10000

Epoch 20 training complete
Cost on training data: 0.319613486199
Accuracy on training data: 48629 / 50000
Cost on evaluation data: 0.789575964369
Accuracy on evaluation data: 9599 / 10000

Epoch 21 training complete
Cost on training data: 0.313774062183
Accuracy on training data: 48675 / 50000
Cost on evaluation data: 0.78846845522
Accuracy on evaluation data: 9623 / 10000

Epoch 22 training complete
```

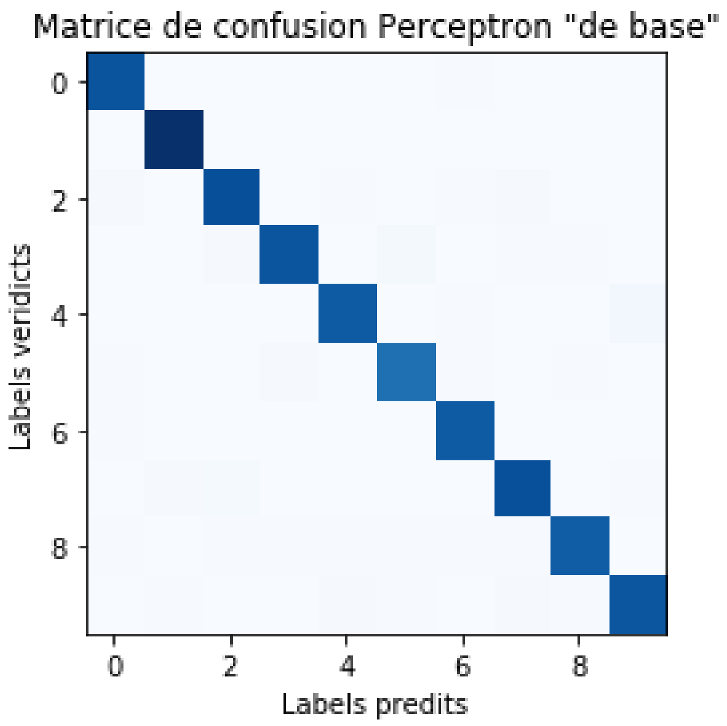
```
Cost on training data: 0.319405337234
Accuracy on training data: 48656 / 50000
Cost on evaluation data: 0.796069046871
Accuracy on evaluation data: 9620 / 10000

Epoch 23 training complete
Cost on training data: 0.319686678332
Accuracy on training data: 48723 / 50000
Cost on evaluation data: 0.799485197319
Accuracy on evaluation data: 9627 / 10000

Epoch 24 training complete
Cost on training data: 0.308342367002
Accuracy on training data: 48771 / 50000
Cost on evaluation data: 0.791090037746
Accuracy on evaluation data: 9637 / 10000

Epoch 25 training complete
Cost on training data: 0.315997539834
Accuracy on training data: 48717 / 50000
Cost on evaluation data: 0.80455805797
Accuracy on evaluation data: 9632 / 10000

Epoch 26 training complete
Cost on training data: 0.314755713013
Accuracy on training data: 48749 / 50000
Cost on evaluation data: 0.802709855979
Accuracy on evaluation data: 9633 / 10000

Epoch 27 training complete
Cost on training data: 0.31209041638
Accuracy on training data: 48750 / 50000
Cost on evaluation data: 0.799329361213
Accuracy on evaluation data: 9629 / 10000

Epoch 28 training complete
Cost on training data: 0.306297292433
Accuracy on training data: 48780 / 50000
Cost on evaluation data: 0.804546293325
Accuracy on evaluation data: 9636 / 10000

Epoch 29 training complete
Cost on training data: 0.305955848359
Accuracy on training data: 48826 / 50000
Cost on evaluation data: 0.80510642729
Accuracy on evaluation data: 9636 / 10000
```

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

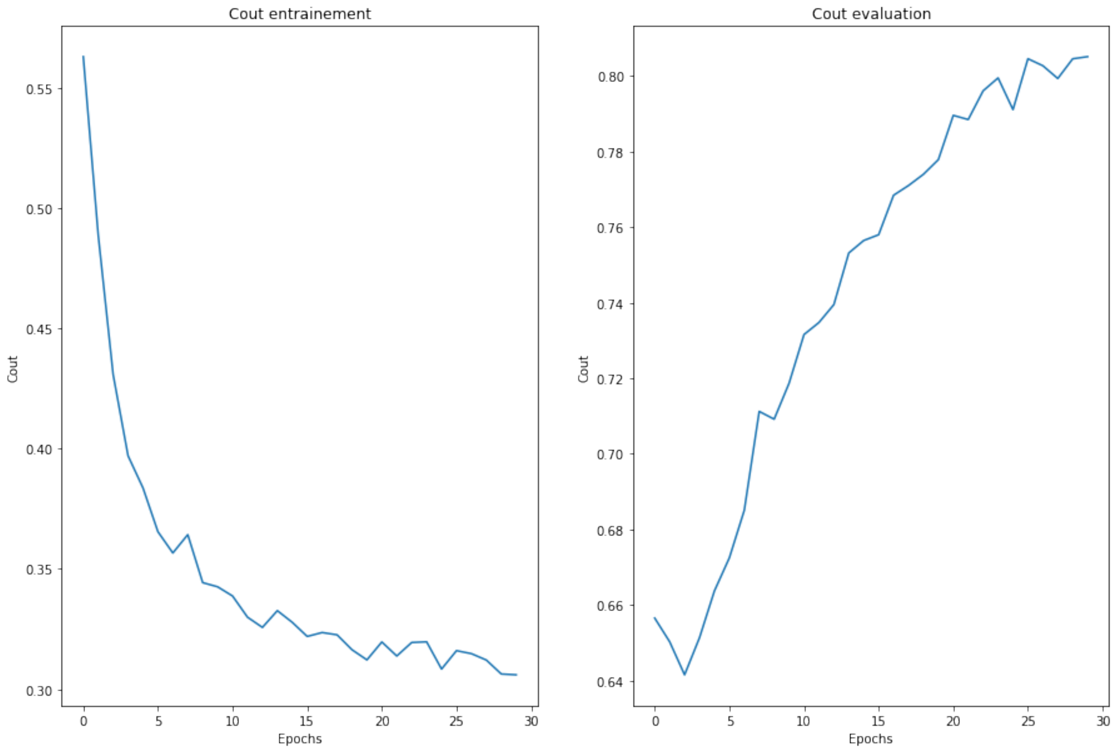
[17]:
```
plot_predict(test_data)
```



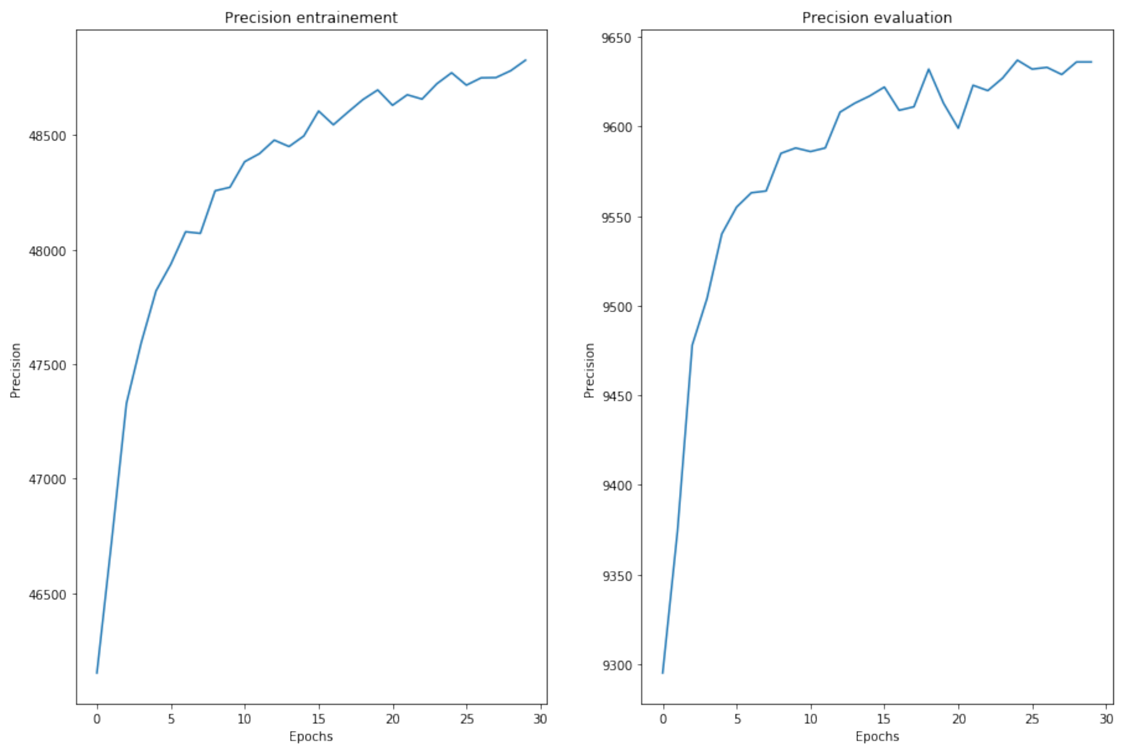
[18]:
```
plot_conf_matrix(test_data)
```



```
[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: 0.54128598241
Accuracy on training data: 46334 / 50000
Cost on evaluation data: 0.63063371219
Accuracy on evaluation data: 9321 / 10000

Epoch 1 training complete
Cost on training data: 0.453994924063
Accuracy on training data: 46996 / 50000
Cost on evaluation data: 0.61487365886
Accuracy on evaluation data: 9424 / 10000
```

```
Epoch 2 training complete
Cost on training data: 0.393617361173
Accuracy on training data: 47584 / 50000
Cost on evaluation data: 0.605840790903
Accuracy on evaluation data: 9559 / 10000

Epoch 3 training complete
Cost on training data: 0.362459795242
Accuracy on training data: 47871 / 50000
Cost on evaluation data: 0.613176804719
Accuracy on evaluation data: 9562 / 10000

Epoch 4 training complete
Cost on training data: 0.344974093115
Accuracy on training data: 48112 / 50000
Cost on evaluation data: 0.62932998211
Accuracy on evaluation data: 9607 / 10000

Epoch 5 training complete
Cost on training data: 0.325891519186
Accuracy on training data: 48329 / 50000
Cost on evaluation data: 0.637066028263
Accuracy on evaluation data: 9621 / 10000

Epoch 6 training complete
Cost on training data: 0.312036912534
Accuracy on training data: 48494 / 50000
Cost on evaluation data: 0.648234798806
Accuracy on evaluation data: 9653 / 10000

Epoch 7 training complete
Cost on training data: 0.30353246859
Accuracy on training data: 48553 / 50000
Cost on evaluation data: 0.658987852
Accuracy on evaluation data: 9662 / 10000

Epoch 8 training complete
Cost on training data: 0.303331760778
Accuracy on training data: 48641 / 50000
Cost on evaluation data: 0.680290791868
Accuracy on evaluation data: 9645 / 10000

Epoch 9 training complete
Cost on training data: 0.288993752316
Accuracy on training data: 48766 / 50000
Cost on evaluation data: 0.678170711514
Accuracy on evaluation data: 9685 / 10000
```

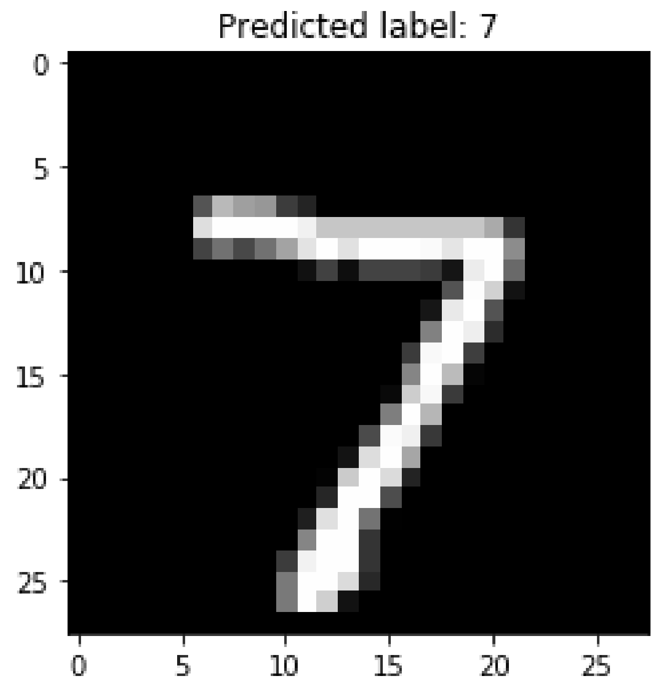
```
Epoch 10 training complete
Cost on training data: 0.284997225192
Accuracy on training data: 48811 / 50000
Cost on evaluation data: 0.68994653989
Accuracy on evaluation data: 9668 / 10000

Epoch 11 training complete
Cost on training data: 0.285457141473
Accuracy on training data: 48885 / 50000
Cost on evaluation data: 0.701380291615
Accuracy on evaluation data: 9689 / 10000

Epoch 12 training complete
Cost on training data: 0.281810680472
Accuracy on training data: 48920 / 50000
Cost on evaluation data: 0.708735260844
Accuracy on evaluation data: 9693 / 10000

Epoch 13 training complete
Cost on training data: 0.277887390243
Accuracy on training data: 48956 / 50000
Cost on evaluation data: 0.718405142787
Accuracy on evaluation data: 9688 / 10000

Epoch 14 training complete
Cost on training data: 0.273348672648
Accuracy on training data: 48989 / 50000
Cost on evaluation data: 0.720614210535
Accuracy on evaluation data: 9695 / 10000

Epoch 15 training complete
Cost on training data: 0.277143172451
Accuracy on training data: 49012 / 50000
Cost on evaluation data: 0.732103172409
Accuracy on evaluation data: 9690 / 10000

Epoch 16 training complete
Cost on training data: 0.270605489902
Accuracy on training data: 49062 / 50000
Cost on evaluation data: 0.732221526579
Accuracy on evaluation data: 9711 / 10000

Epoch 17 training complete
Cost on training data: 0.27636447165
Accuracy on training data: 49043 / 50000
Cost on evaluation data: 0.742972670397
Accuracy on evaluation data: 9697 / 10000
```

```
Epoch 18 training complete
Cost on training data: 0.266953133817
Accuracy on training data: 49112 / 50000
Cost on evaluation data: 0.743716504546
Accuracy on evaluation data: 9702 / 10000

Epoch 19 training complete
Cost on training data: 0.264355185204
Accuracy on training data: 49141 / 50000
Cost on evaluation data: 0.748137813779
Accuracy on evaluation data: 9706 / 10000

Epoch 20 training complete
Cost on training data: 0.271833454824
Accuracy on training data: 49117 / 50000
Cost on evaluation data: 0.758872798909
Accuracy on evaluation data: 9711 / 10000

Epoch 21 training complete
Cost on training data: 0.267378859008
Accuracy on training data: 49212 / 50000
Cost on evaluation data: 0.759496350041
Accuracy on evaluation data: 9726 / 10000

Epoch 22 training complete
Cost on training data: 0.26479052225
Accuracy on training data: 49164 / 50000
Cost on evaluation data: 0.76056305445
Accuracy on evaluation data: 9712 / 10000

Epoch 23 training complete
Cost on training data: 0.262844284548
Accuracy on training data: 49194 / 50000
Cost on evaluation data: 0.765252949763
Accuracy on evaluation data: 9706 / 10000

Epoch 24 training complete
Cost on training data: 0.259852657898
Accuracy on training data: 49210 / 50000
Cost on evaluation data: 0.766177600755
Accuracy on evaluation data: 9727 / 10000

Epoch 25 training complete
Cost on training data: 0.2573240465
Accuracy on training data: 49240 / 50000
Cost on evaluation data: 0.767883635407
Accuracy on evaluation data: 9713 / 10000
```

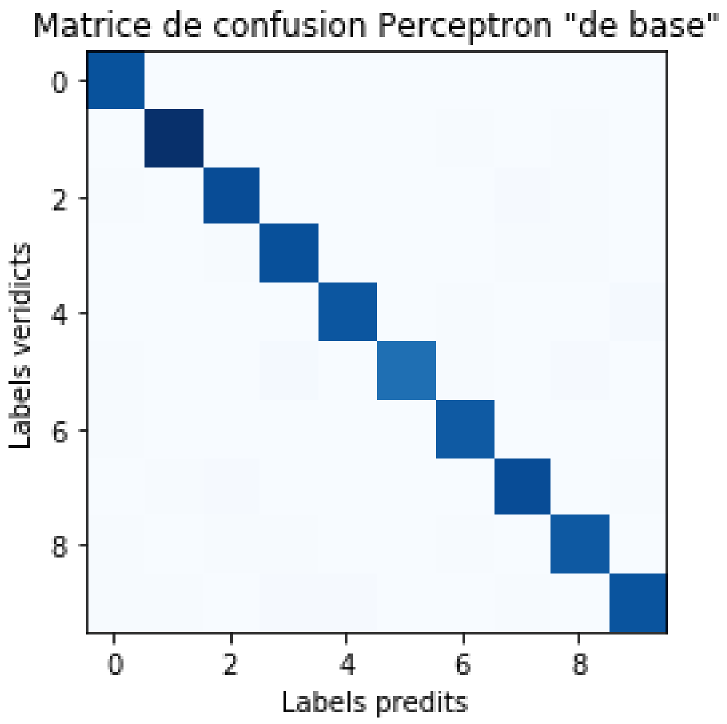
```
Epoch 26 training complete
Cost on training data: 0.258755104621
Accuracy on training data: 49263 / 50000
Cost on evaluation data: 0.772940095448
Accuracy on evaluation data: 9729 / 10000

Epoch 27 training complete
Cost on training data: 0.256249706791
Accuracy on training data: 49256 / 50000
Cost on evaluation data: 0.772992956341
Accuracy on evaluation data: 9722 / 10000

Epoch 28 training complete
Cost on training data: 0.258802717066
Accuracy on training data: 49291 / 50000
Cost on evaluation data: 0.777247990448
Accuracy on evaluation data: 9718 / 10000

Epoch 29 training complete
Cost on training data: 0.258788980105
Accuracy on training data: 49279 / 50000
Cost on evaluation data: 0.782893857999
Accuracy on evaluation data: 9716 / 10000
```

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

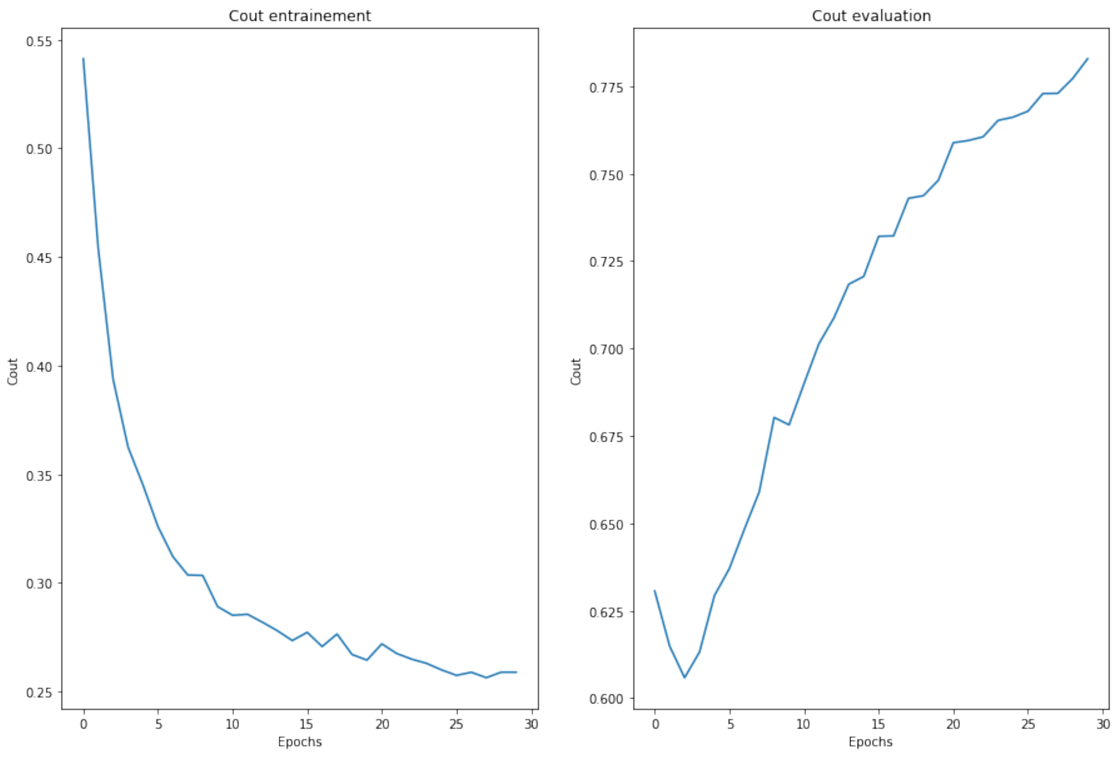
[24]:
```
plot_predict(test_data)
```



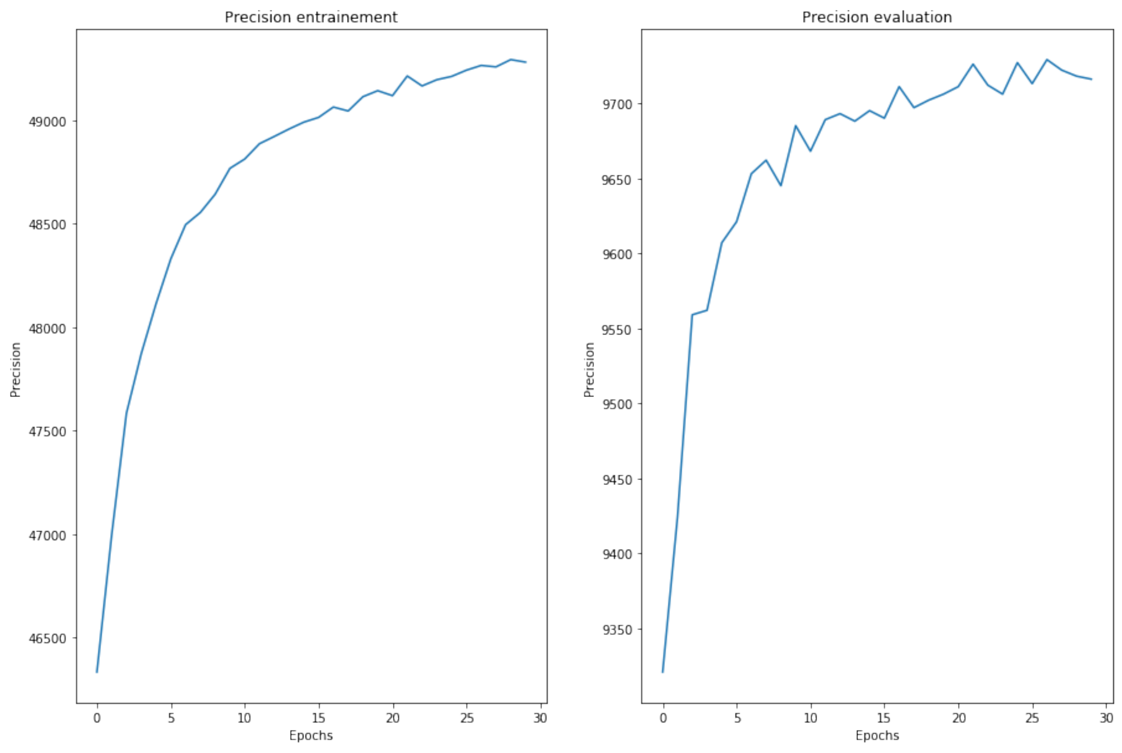
[25]:
```
plot_conf_matrix(test_data)
```



[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: 3.26230906442
Accuracy on training data: 5678 / 50000
Cost on evaluation data: 3.2937666632
Accuracy on evaluation data: 1064 / 10000

Epoch 1 training complete
Cost on training data: 2.02942318955
Accuracy on training data: 25006 / 50000
Cost on evaluation data: 2.12468005432
Accuracy on evaluation data: 5069 / 10000

Epoch 2 training complete
Cost on training data: 0.834494345975
Accuracy on training data: 44523 / 50000
Cost on evaluation data: 1.072576563
Accuracy on evaluation data: 8972 / 10000

Epoch 3 training complete
Cost on training data: 0.625530001095
Accuracy on training data: 45985 / 50000
Cost on evaluation data: 0.975507775456
Accuracy on evaluation data: 9184 / 10000

Epoch 4 training complete
Cost on training data: 0.541235413998
Accuracy on training data: 46727 / 50000
Cost on evaluation data: 0.975059996059
Accuracy on evaluation data: 9328 / 10000

Epoch 5 training complete
Cost on training data: 0.478953175849
Accuracy on training data: 47190 / 50000
Cost on evaluation data: 0.9622131684
Accuracy on evaluation data: 9386 / 10000

Epoch 6 training complete
Cost on training data: 0.428828682994
Accuracy on training data: 47724 / 50000
Cost on evaluation data: 0.930228851203
```

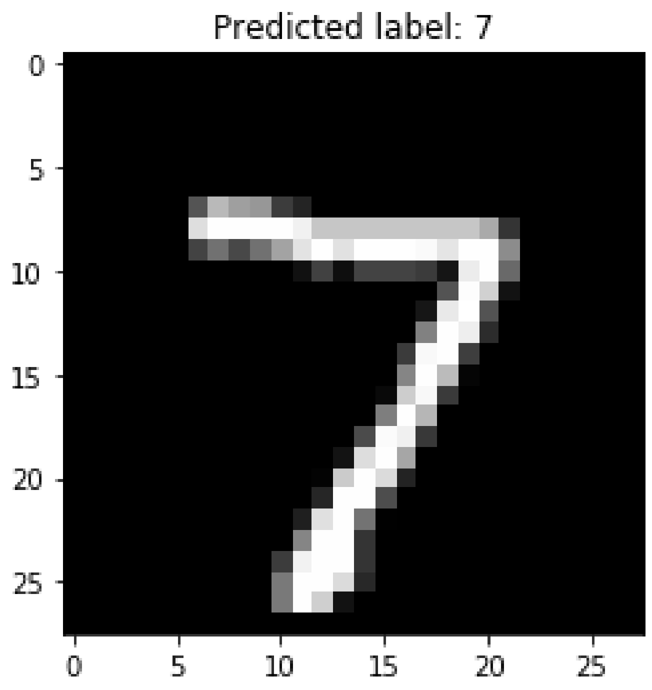
```
Accuracy on evaluation data: 9493 / 10000

Epoch 7 training complete
Cost on training data: 0.4243871344
Accuracy on training data: 47664 / 50000
Cost on evaluation data: 0.945960330367
Accuracy on evaluation data: 9461 / 10000

Epoch 8 training complete
Cost on training data: 0.393944643615
Accuracy on training data: 48025 / 50000
Cost on evaluation data: 0.938340411347
Accuracy on evaluation data: 9517 / 10000

Epoch 9 training complete
Cost on training data: 0.368812111971
Accuracy on training data: 48183 / 50000
Cost on evaluation data: 0.936021696168
Accuracy on evaluation data: 9546 / 10000

Epoch 10 training complete
Cost on training data: 0.355449833607
Accuracy on training data: 48376 / 50000
Cost on evaluation data: 0.936290731432
Accuracy on evaluation data: 9568 / 10000

Epoch 11 training complete
Cost on training data: 0.347210520464
Accuracy on training data: 48454 / 50000
Cost on evaluation data: 0.947102555081
Accuracy on evaluation data: 9565 / 10000

Epoch 12 training complete
Cost on training data: 0.336080701597
Accuracy on training data: 48540 / 50000
Cost on evaluation data: 0.94071842849
Accuracy on evaluation data: 9586 / 10000

Epoch 13 training complete
Cost on training data: 0.342927947516
Accuracy on training data: 48511 / 50000
Cost on evaluation data: 0.951142955884
Accuracy on evaluation data: 9596 / 10000

Epoch 14 training complete
Cost on training data: 0.361972227178
Accuracy on training data: 48277 / 50000
Cost on evaluation data: 0.979590216395
```

```
Accuracy on evaluation data: 9538 / 10000

Epoch 15 training complete
Cost on training data: 0.358158503388
Accuracy on training data: 48304 / 50000
Cost on evaluation data: 0.994274012155
Accuracy on evaluation data: 9546 / 10000

Epoch 16 training complete
Cost on training data: 0.314687362088
Accuracy on training data: 48736 / 50000
Cost on evaluation data: 0.949108291326
Accuracy on evaluation data: 9611 / 10000

Epoch 17 training complete
Cost on training data: 0.323549426481
Accuracy on training data: 48673 / 50000
Cost on evaluation data: 0.958417499892
Accuracy on evaluation data: 9601 / 10000

Epoch 18 training complete
Cost on training data: 0.304943993675
Accuracy on training data: 48814 / 50000
Cost on evaluation data: 0.946210842769
Accuracy on evaluation data: 9623 / 10000

Epoch 19 training complete
Cost on training data: 0.311810583345
Accuracy on training data: 48739 / 50000
Cost on evaluation data: 0.96335253765
Accuracy on evaluation data: 9626 / 10000

Epoch 20 training complete
Cost on training data: 0.332570656878
Accuracy on training data: 48584 / 50000
Cost on evaluation data: 0.980211533447
Accuracy on evaluation data: 9613 / 10000

Epoch 21 training complete
Cost on training data: 0.283387402242
Accuracy on training data: 49036 / 50000
Cost on evaluation data: 0.945236223837
Accuracy on evaluation data: 9654 / 10000

Epoch 22 training complete
Cost on training data: 0.304954889158
Accuracy on training data: 48801 / 50000
Cost on evaluation data: 0.970032068123
```

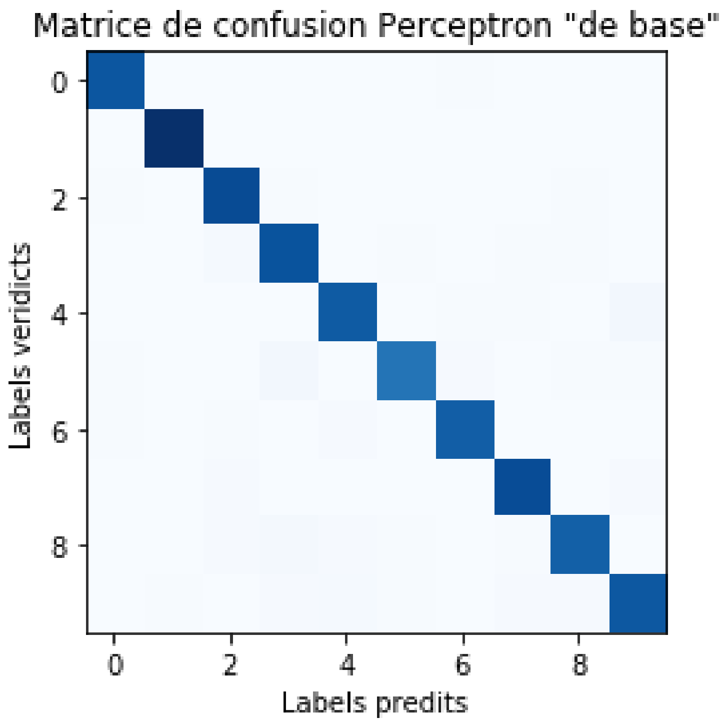
```
Accuracy on evaluation data: 9624 / 10000

Epoch 23 training complete
Cost on training data: 0.304884389956
Accuracy on training data: 48815 / 50000
Cost on evaluation data: 0.983608358044
Accuracy on evaluation data: 9627 / 10000

Epoch 24 training complete
Cost on training data: 0.321071091219
Accuracy on training data: 48661 / 50000
Cost on evaluation data: 0.994542752881
Accuracy on evaluation data: 9587 / 10000

Epoch 25 training complete
Cost on training data: 0.279061038624
Accuracy on training data: 49060 / 50000
Cost on evaluation data: 0.954179861792
Accuracy on evaluation data: 9645 / 10000

Epoch 26 training complete
Cost on training data: 0.296905184945
Accuracy on training data: 48864 / 50000
Cost on evaluation data: 0.969702163583
Accuracy on evaluation data: 9635 / 10000

Epoch 27 training complete
Cost on training data: 0.324053928255
Accuracy on training data: 48625 / 50000
Cost on evaluation data: 0.998920318247
Accuracy on evaluation data: 9579 / 10000

Epoch 28 training complete
Cost on training data: 0.274027358888
Accuracy on training data: 49101 / 50000
Cost on evaluation data: 0.964603203097
Accuracy on evaluation data: 9656 / 10000

Epoch 29 training complete
Cost on training data: 0.269688957894
Accuracy on training data: 49138 / 50000
Cost on evaluation data: 0.958555224294
Accuracy on evaluation data: 9665 / 10000
```

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

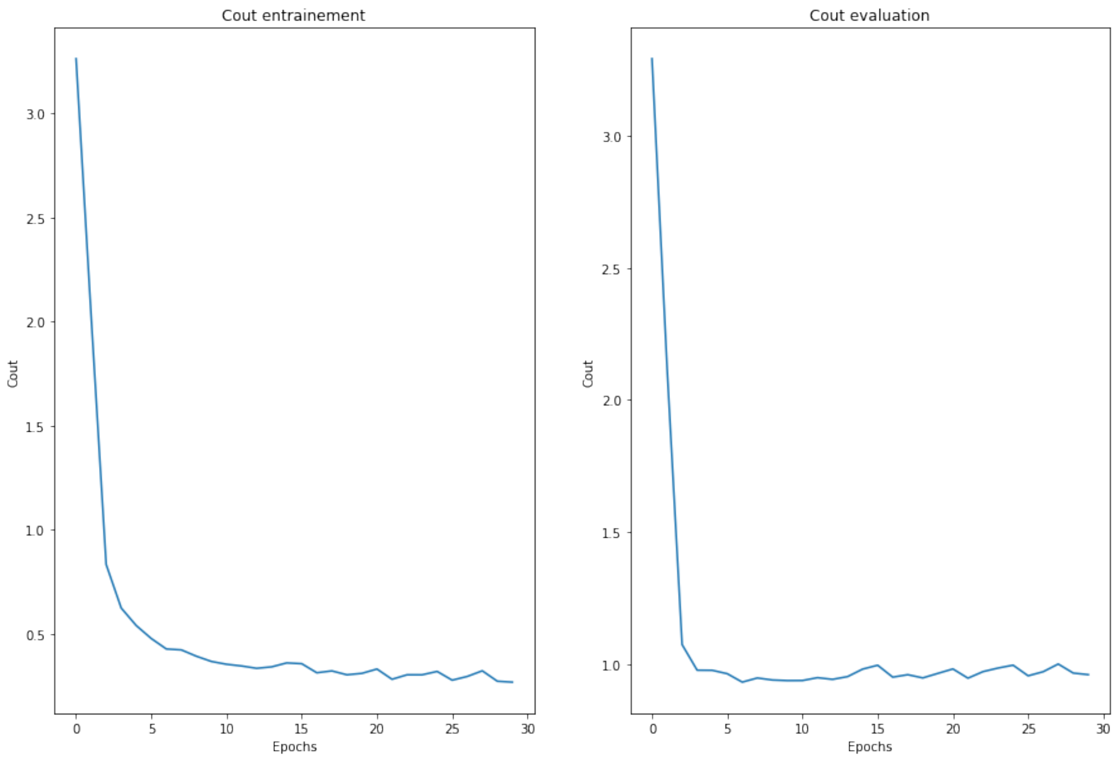
[32]:
```
plot_predict(test_data)
```



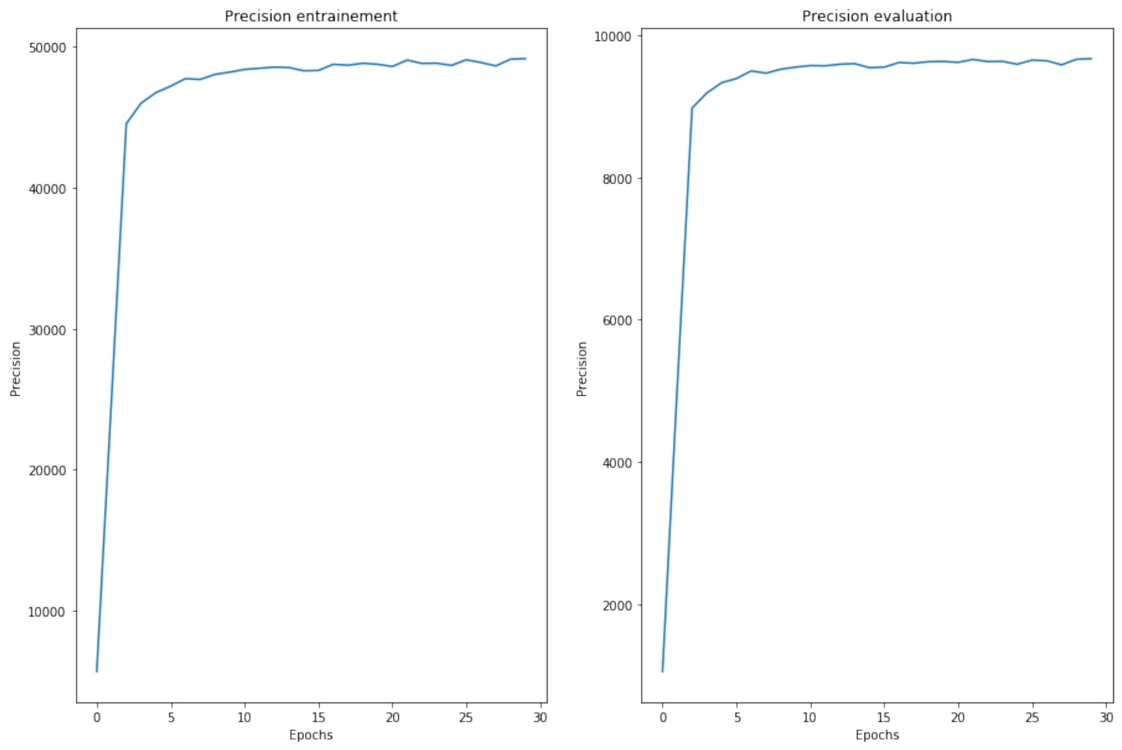
[33]:
```
plot_conf_matrix(test_data)
```



[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

[102]:
```
import network2
import numpy as np
import mnist_loader
import matplotlib.pyplot as plt
from sklearn.metrics import confusion_matrix
```

[103]:
```
training_data, validation_data, test_data = mnist_loader.load_data_wrapper()
training_data = list(training_data)
```

[104]:
```
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: 3.27334571818
Accuracy on training data: 5678 / 50000
Cost on evaluation data: 3.33315218871
Accuracy on evaluation data: 1064 / 10000

Epoch 1 training complete
Cost on training data: 3.26651231987
Accuracy on training data: 4951 / 50000
Cost on evaluation data: 3.32046107732
Accuracy on evaluation data: 967 / 10000

Epoch 2 training complete
Cost on training data: 3.26594775262
Accuracy on training data: 5678 / 50000
Cost on evaluation data: 3.31860638794
Accuracy on evaluation data: 1064 / 10000

Epoch 3 training complete
Cost on training data: 3.26533987298
Accuracy on training data: 5678 / 50000
Cost on evaluation data: 3.31247162176
Accuracy on evaluation data: 1064 / 10000

Epoch 4 training complete
Cost on training data: 3.26574107786
Accuracy on training data: 5175 / 50000
Cost on evaluation data: 3.30746296624
Accuracy on evaluation data: 1090 / 10000

Epoch 5 training complete
Cost on training data: 3.26120432135
Accuracy on training data: 5678 / 50000
Cost on evaluation data: 3.30103207735
Accuracy on evaluation data: 1064 / 10000

Epoch 6 training complete
```

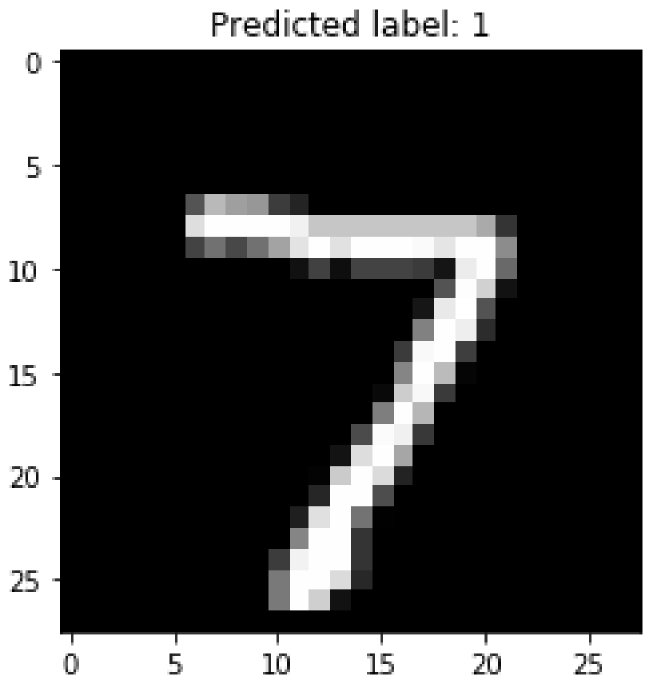
```
Cost on training data: 3.26234502895
Accuracy on training data: 4988 / 50000
Cost on evaluation data: 3.299837075
Accuracy on evaluation data: 961 / 10000

Epoch 7 training complete
Cost on training data: 3.25997541192
Accuracy on training data: 4951 / 50000
Cost on evaluation data: 3.29389613098
Accuracy on evaluation data: 967 / 10000

Epoch 8 training complete
Cost on training data: 3.25939674615
Accuracy on training data: 4859 / 50000
Cost on evaluation data: 3.28983994264
Accuracy on evaluation data: 983 / 10000

Epoch 9 training complete
Cost on training data: 3.25836788774
Accuracy on training data: 5678 / 50000
Cost on evaluation data: 3.28771235521
Accuracy on evaluation data: 1064 / 10000

Epoch 10 training complete
Cost on training data: 3.25924365227
Accuracy on training data: 5175 / 50000
Cost on evaluation data: 3.283800951
Accuracy on evaluation data: 1090 / 10000

Epoch 11 training complete
Cost on training data: 3.25626340264
Accuracy on training data: 5678 / 50000
Cost on evaluation data: 3.2815101983
Accuracy on evaluation data: 1064 / 10000

Epoch 12 training complete
Cost on training data: 3.25630078487
Accuracy on training data: 5101 / 50000
Cost on evaluation data: 3.27870064707
Accuracy on evaluation data: 1030 / 10000

Epoch 13 training complete
Cost on training data: 3.25777330143
Accuracy on training data: 5678 / 50000
Cost on evaluation data: 3.281175644
Accuracy on evaluation data: 1064 / 10000

Epoch 14 training complete
```

```
Cost on training data: 3.25508542356
Accuracy on training data: 5678 / 50000
Cost on evaluation data: 3.27386578881
Accuracy on evaluation data: 1064 / 10000

Epoch 15 training complete
Cost on training data: 3.25465809913
Accuracy on training data: 4951 / 50000
Cost on evaluation data: 3.27275293451
Accuracy on evaluation data: 967 / 10000

Epoch 16 training complete
Cost on training data: 3.25420126369
Accuracy on training data: 5101 / 50000
Cost on evaluation data: 3.26989914931
Accuracy on evaluation data: 1030 / 10000

Epoch 17 training complete
Cost on training data: 3.25556338987
Accuracy on training data: 4932 / 50000
Cost on evaluation data: 3.27091115208
Accuracy on evaluation data: 991 / 10000

Epoch 18 training complete
Cost on training data: 3.25428043071
Accuracy on training data: 5678 / 50000
Cost on evaluation data: 3.26962765879
Accuracy on evaluation data: 1064 / 10000

Epoch 19 training complete
Cost on training data: 3.25359179632
Accuracy on training data: 5678 / 50000
Cost on evaluation data: 3.26753233032
Accuracy on evaluation data: 1064 / 10000

Epoch 20 training complete
Cost on training data: 3.25455252521
Accuracy on training data: 4988 / 50000
Cost on evaluation data: 3.26779614959
Accuracy on evaluation data: 961 / 10000

Epoch 21 training complete
Cost on training data: 3.25491788947
Accuracy on training data: 5678 / 50000
Cost on evaluation data: 3.26741365244
Accuracy on evaluation data: 1064 / 10000

Epoch 22 training complete
```

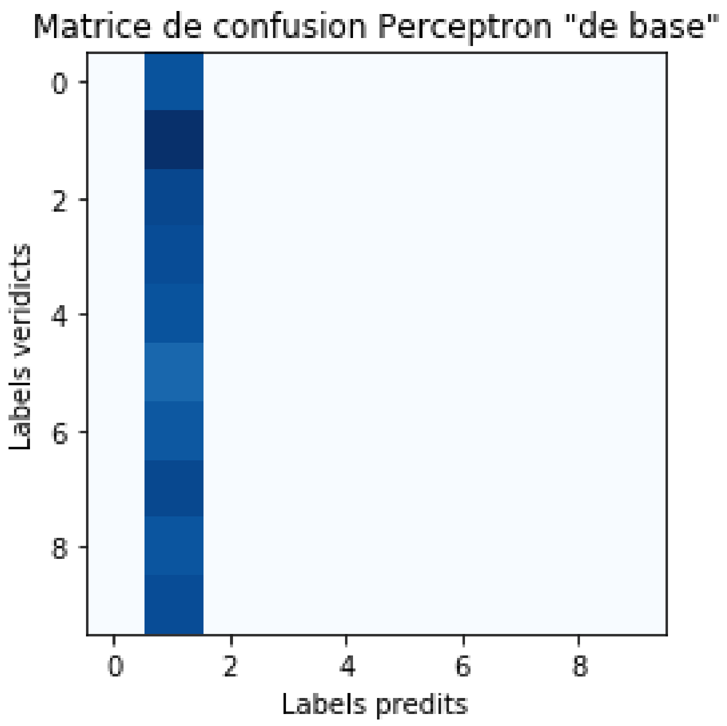
```
Cost on training data: 3.25362093629
Accuracy on training data: 5678 / 50000
Cost on evaluation data: 3.26447440985
Accuracy on evaluation data: 1064 / 10000

Epoch 23 training complete
Cost on training data: 3.25287285897
Accuracy on training data: 5678 / 50000
Cost on evaluation data: 3.26457929125
Accuracy on evaluation data: 1064 / 10000

Epoch 24 training complete
Cost on training data: 3.25643610485
Accuracy on training data: 5175 / 50000
Cost on evaluation data: 3.26372961002
Accuracy on evaluation data: 1090 / 10000

Epoch 25 training complete
Cost on training data: 3.25288375512
Accuracy on training data: 5678 / 50000
Cost on evaluation data: 3.26316044856
Accuracy on evaluation data: 1064 / 10000

Epoch 26 training complete
Cost on training data: 3.25358781782
Accuracy on training data: 4932 / 50000
Cost on evaluation data: 3.26130168938
Accuracy on evaluation data: 991 / 10000

Epoch 27 training complete
Cost on training data: 3.25229689057
Accuracy on training data: 5678 / 50000
Cost on evaluation data: 3.26102923317
Accuracy on evaluation data: 1064 / 10000

Epoch 28 training complete
Cost on training data: 3.25346255763
Accuracy on training data: 5678 / 50000
Cost on evaluation data: 3.26291693475
Accuracy on evaluation data: 1064 / 10000

Epoch 29 training complete
Cost on training data: 3.25470882334
Accuracy on training data: 5678 / 50000
Cost on evaluation data: 3.26203449635
Accuracy on evaluation data: 1064 / 10000
```

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

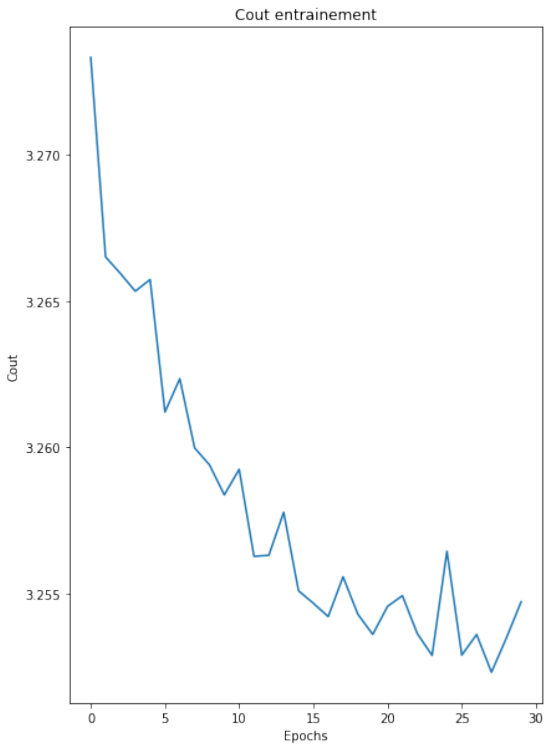
```
[106]: plot_predict(test_data)
```

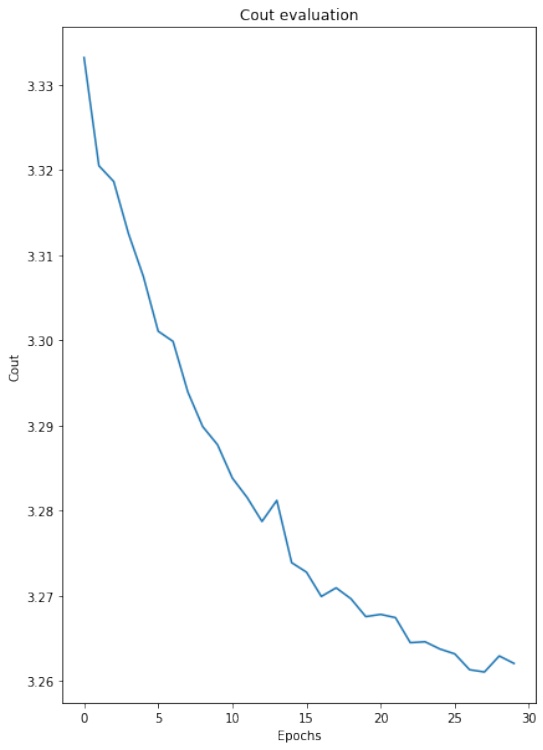


```
[107]: plot_conf_matrix(test_data)
```

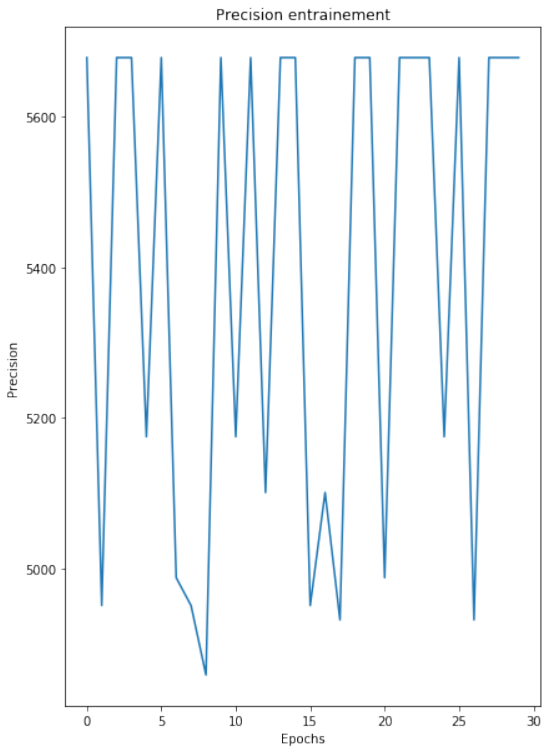


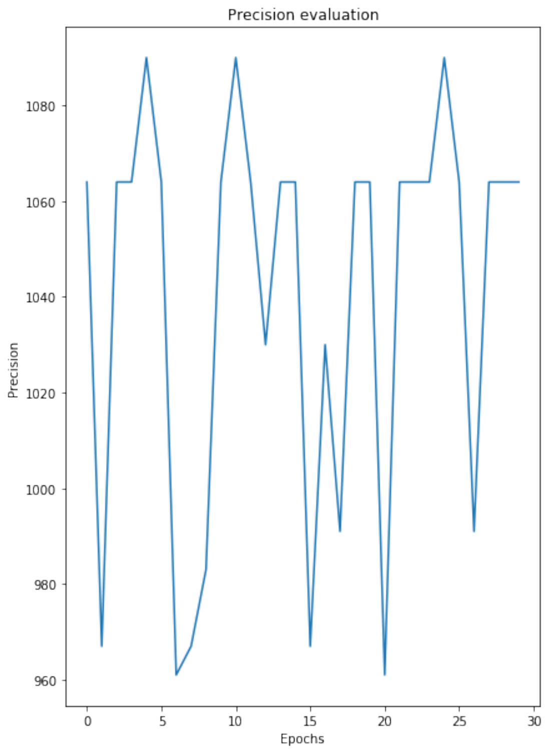
[108]:
```
subplots_data(
    [
        {
            "data": training_cost,
            "plot_tile": 'Cout entrainement',
            "label": ('Epochs', 'Cout'),
        },
        {
            "data": evaluation_cost,
            "plot_tile": 'Cout evaluation',
            "label": ('Epochs', 'Cout'),
        }
    ],
    1,
    2
)
```





[109]:
```
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

[52]:
```python
import network3
from network3 import ConvPoolLayer, FullyConnectedLayer, SoftmaxLayer, ReLU
import numpy as np
import matplotlib.pyplot as plt
```

[53]:
```python
training_data, validation_data, test_data = network3.load_data_shared()
```

[54]:
```python
epochs = 30
mini_batch_size = 10
eta = 0.03
```

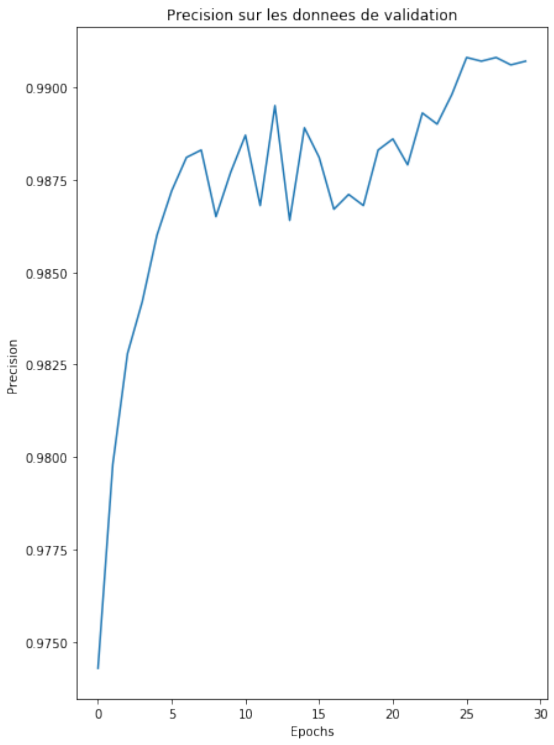
[55]:
```python
net = network3.Network([
    ConvPoolLayer(image_shape=(mini_batch_size, 1, 28, 28),
                  filter_shape=(20, 1, 5, 5),
                  poolsize=(2, 2),
                  activation_fn=ReLU),
    ConvPoolLayer(image_shape=(mini_batch_size, 20, 12, 12),
                  filter_shape=(40, 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)
```

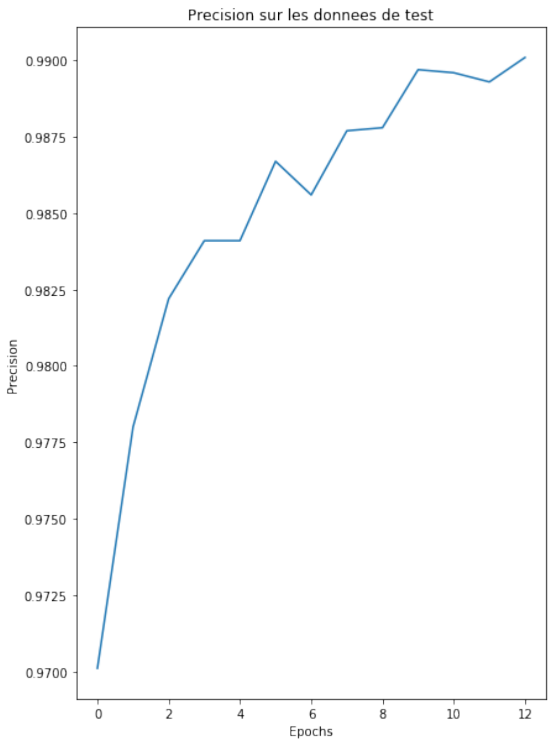
```
Training mini-batch number 0
Training mini-batch number 1000
Training mini-batch number 2000
Training mini-batch number 3000
Training mini-batch number 4000
Epoch 0: validation accuracy 97.43%
This is the best validation accuracy to date.
The corresponding test accuracy is 97.01%
Training mini-batch 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 97.98%
This is the best validation accuracy to date.
The corresponding test accuracy is 97.80%
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 98.28%
This is the best validation accuracy to date.
The corresponding test accuracy is 98.22%
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 98.42%
This is the best validation accuracy to date.
The corresponding test accuracy is 98.41%
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 98.60%
This is the best validation accuracy to date.
The corresponding test accuracy is 98.41%
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 98.72%
This is the best validation accuracy to date.
The corresponding test accuracy is 98.67%
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 98.81%
This is the best validation accuracy to date.
The corresponding test accuracy is 98.56%
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 98.83%
This is the best validation accuracy to date.
The corresponding test accuracy is 98.77%
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 98.65%
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 98.77%
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 98.87%
This is the best validation accuracy to date.
The corresponding test accuracy is 98.78%
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 98.68%
```

```
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 98.95%
This is the best validation accuracy to date.
The corresponding test accuracy is 98.97%
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 98.64%
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 98.89%
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 98.81%
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 98.67%
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 98.71%
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 98.68%
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 98.83%
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 98.86%
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 98.79%
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 98.93%
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 98.90%
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 98.98%
This is the best validation accuracy to date.
The corresponding test accuracy is 98.96%
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 99.08%
This is the best validation accuracy to date.
The corresponding test accuracy is 98.93%
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 99.07%
```

```
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 99.08%
This is the best validation accuracy to date.
The corresponding test accuracy is 99.01%
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 99.08% obtained at iteration 139999
Corresponding test accuracy of 99.01%
```

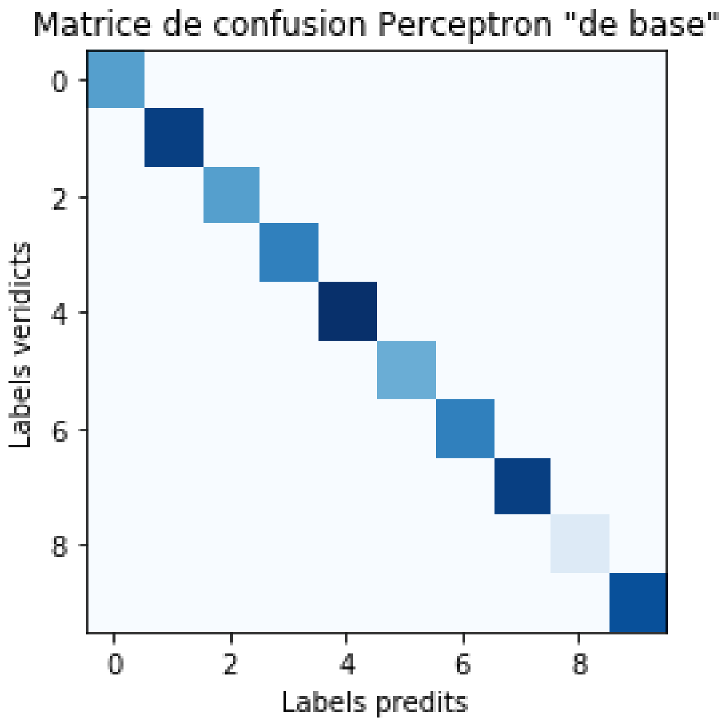
[58]:
```
subplots_data(
    [
        {
            "data": validation_accuracy,
            "plot_tile": 'Precision sur les donnees de validation',
            "label": ('Epochs', 'Precision'),
        },
        {
            "data": test_accuracy,
            "plot_tile": 'Precision sur les donnees de test',
            "label": ('Epochs', 'Precision'),
        }
    ],
    1,
    2
)
```





[95]:
```
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)])
```

[96]:
```
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 Perceptron \"de base\"")
plt.show()
```


Matrice de confusion Perceptron "de base"
Labels veridicts
Labels predits
0
2
4
6
8