TensorFlow Lite : Google pousse un peu plus le machine learning en périphérie

Comment favoriser l'exploitation de modèles d'apprentissage automatique en périphérie du réseau ?

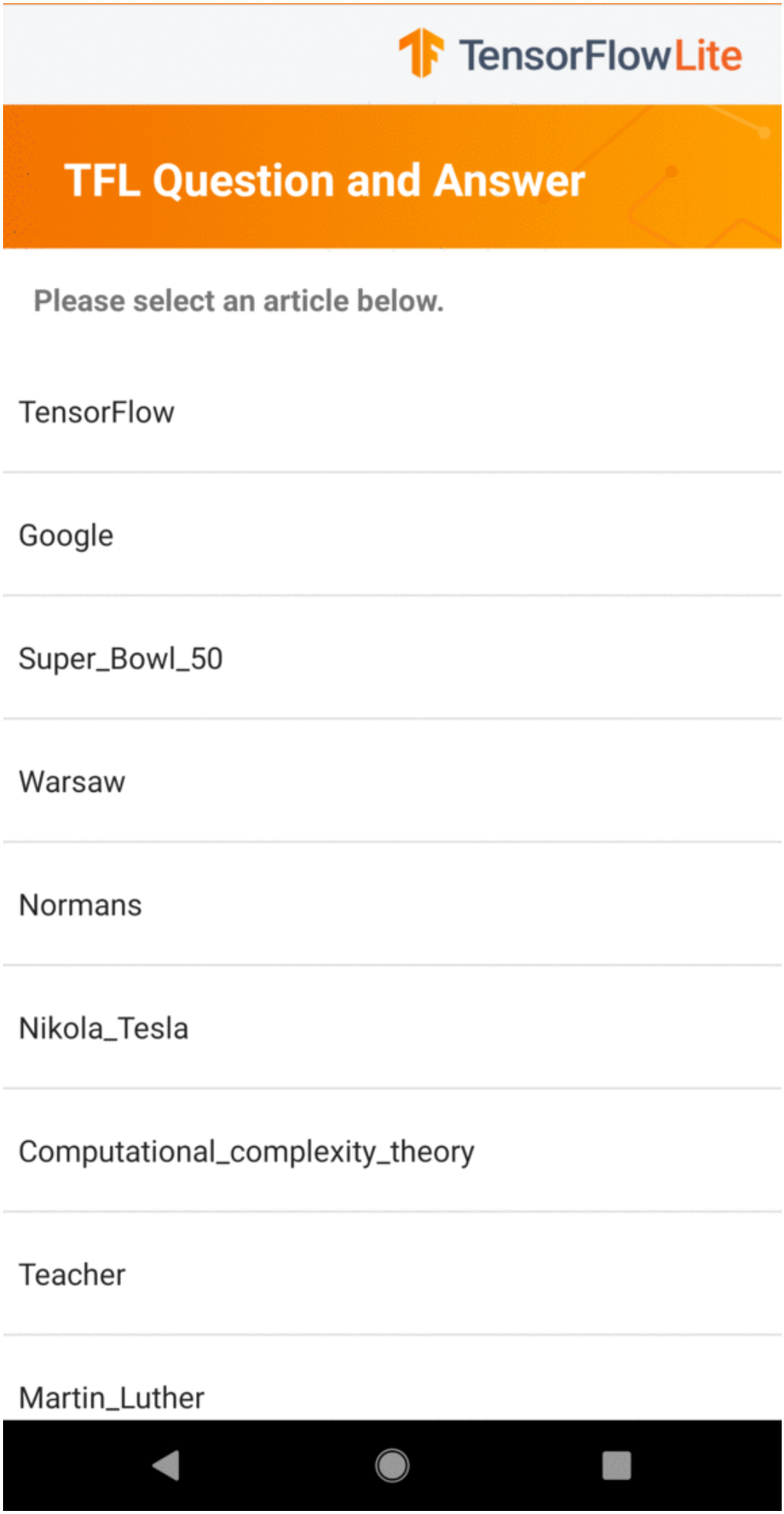
Pour répondre à cette question, Google a lancé, voilà un peu plus de deux ans, une déclinaison « Lite » de son *framework* TensorFlow. Elle cible plus particulièrement les terminaux mobiles (iOS, Android) et les systèmes Linux embarqués.

Il en a été longuement question le mois dernier à l'occasion du TensorFlow Dev Summit (cf. vidéo ci-dessous).

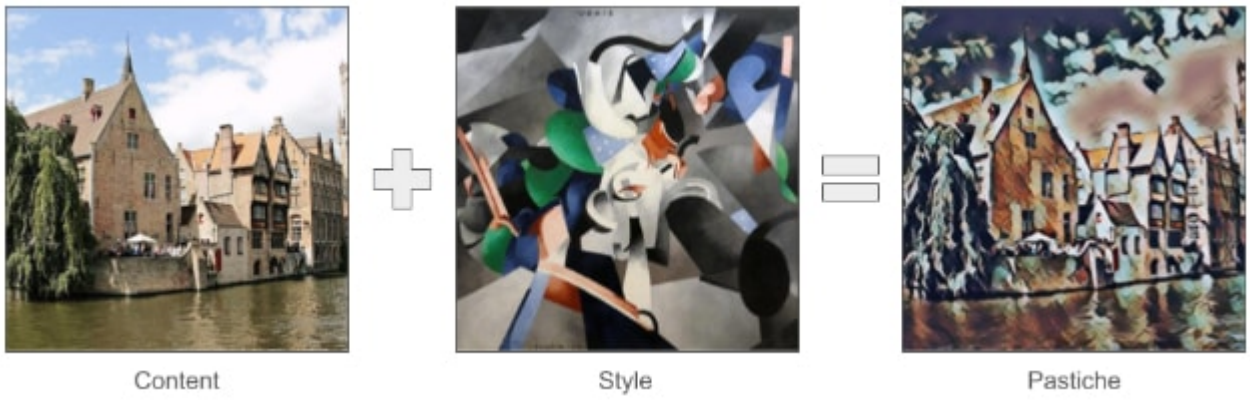
Google en a tiré une synthèse, sous un angle : « du prototype à la production ».

Au premier stade, l'accompagnement se fait notamment à travers un catalogue de modèles « prêts à l'emploi ». Parmi les derniers ajouts, on aura relevé :

- Un modèle fondé sur BERT et affiné sur la base du jeu de données SQuAD (Stanford Question Answering Dataset). Il permet de répondre à des questions à partir de l'analyse d'un texte.

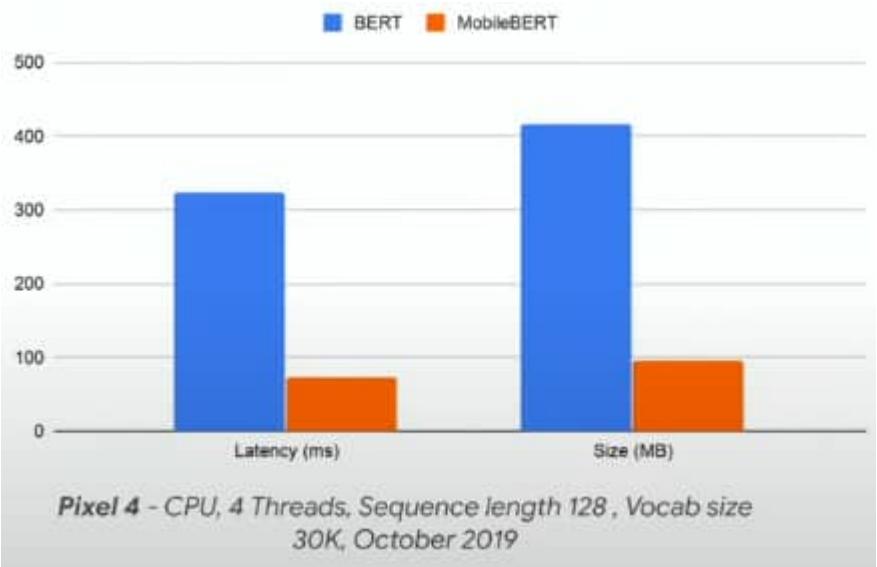

TensorFlowLite
TFL Question and Answer
Please select an article below.
TensorFlow
Google
Super_Bowl_50
Warsaw
Normans
Nikola_Tesla
Computational_complexity_theory
Teacher
Martin_Luther

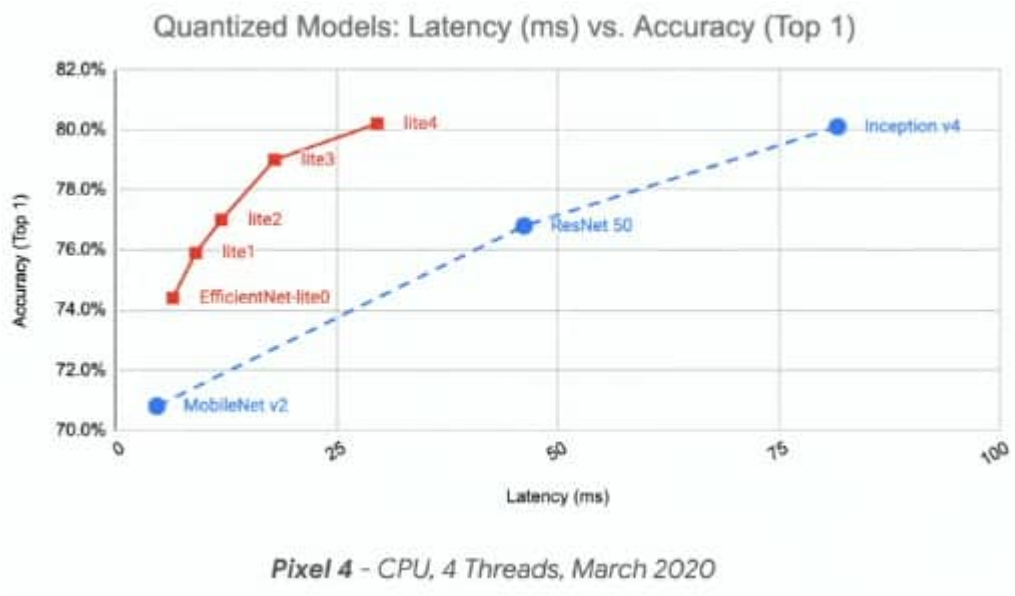
- Un autre de type « transfert de style » qui adapte une image à partir d'un modèle artistique



Ces modèles – et d'autres présents au catalogue – s'assortissent d'applications qui les implémentent.

En parallèle, TensorFlow s'ouvre à des modèles issus de la communauté. Parmi eux, MobileBERT (version de BERT optimisée pour les terminaux mobiles) et Efficient-Lite (pour la classification d'images, avec prise en charge de la quantisation).





Pixel 4 - CPU, 4 Threads, March 2020

Cliquez, c'est intégré ?

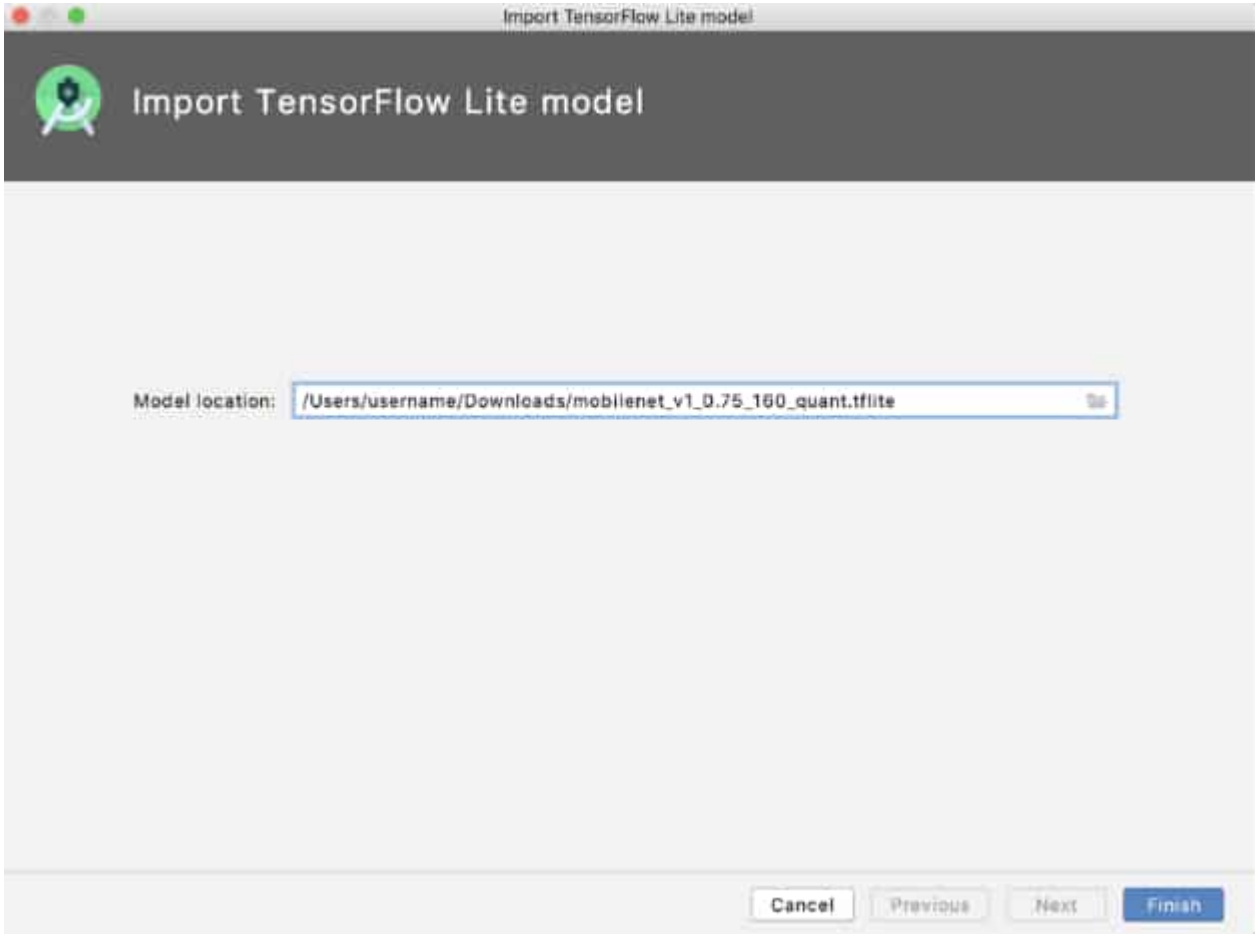
Autre nouveauté : Model Maker.

Cette bibliothèque logicielle doit faciliter l'adaptation des modèles à des jeux de données personnalisés lors de leur déploiement en *edge*. Deux cas d'usage sont pour le moment pris en charge : la classification de texte et d'images. Il est prévu d'y ajouter prochainement la compatibilité avec BERT pour des tâches d'interprétation du langage naturel.

```
# 1. Load data.
data = ImageClassifierDataLoader.from_folder('flower_photos/')

# 2. Customize the model.
model = image_classifier.create(data) # Default model is EfficientNet-Lite0

# 3. Evaluate the model.
loss, accuracy = model.evaluate()

# 4. Export to tflite.
model.export('flower_classifier.tflite')
```

Model Maker implémente les métadonnées « enrichies » que Google a commencé à mettre en place pour simplifier le partage de modèles entre leurs concepteurs et les développeurs.

Ces métadonnées alimentent entre autres un générateur de code qui transforme les modèles en *wrappers* intégrables aux applications Android.
Prochaine étape : étendre cette expérience à Android Studio. Un aperçu est disponible sur le canal canary pour la classification d'images.



À noter également, des évolutions pour les outils de *benchmark* de TensorFlow Lite. Entre autres, l'extension des options d'exécution (spécification du nombre de threads, évaluation de la latence opération par opération…).

Par ailleurs, la liste des plates-formes d'accélération hardware prises en charge s'allonge, avec :

- Sur Android, les DSP Qualcomm Hexagon
- Sur iOS, les puces Neural Engine d'Apple

Illustrations © Google