A képen szöveg, Betűtípus, képernyőkép, embléma látható

Automatikusan generált leírás

Mesterséges intelligencia

**Pytorch Image Classification**

**Deep Learning-el**

**Foodifier**

Ételtípusok felismerése képről a Food 101 dataset alapján

**Kálmán Dávid**

**LNF8KS**

2023/2024/2

Tartalom

[A probléma és a projekt leírása 3](#_Toc166186459)

[Használt technológiák 4](#_Toc166186460)

[Image 101 Dataset 4](#_Toc166186461)

[PyTorch 4](#_Toc166186462)

[fast.ai 4](#_Toc166186463)

[ResNet 4](#_Toc166186464)

[Használati útmutató 5](#_Toc166186465)

[Github Repository 5](#_Toc166186466)

[Telepítési útmutató 5](#_Toc166186467)

[Használata fájlból 5](#_Toc166186468)

[A program bemutatása 6](#_Toc166186469)

[Szerkezeti felépítés 6](#_Toc166186470)

[A model elkészítése 6](#_Toc166186471)

[A model finomítása 8](#_Toc166186472)

[A model használata 8](#_Toc166186473)

[Irodalomjegyzék 9](#_Toc166186474)

## A probléma és a projekt leírása

A projektem célja az volt, hogy egy mesterséges intelligencia alapú (jelen esetben deep learninget alkalmazó) programot készítsek, amely alkalmas megállapítani azt, hogy egy, a programba feltöltött képen milyen típusú étel szerepel.

Azért választottam ezt a témát, mert mindig is érdekelt a képfelismerés és az ehhez tartozó technológiai megoldások. A fejlesztés során a PyTorch gépi tanulási könyvtárat használtam, a modell képzése pedig ResNet-tel történt (több verziójával is próbáltam).

A témaválasztásomban leírt 4 db ételtípus helyett a végső verzió 48 darab típust tud felismerni, 0.33-as hibaértékkel. Ez a gyakorlatban azt jelenti, hogy az esetek 67%-ban sikeresen meg tudja mondani, hogy ténylegesen milyen étel szerepel a képen. Természetesen ennek alap feltétele az, hogy a tesztelt fotón egy olyan ételtípus legyen, amely benne van a 48 darab betanított típusban, különben a program a hozzá leginkább hasonlító formát fogja megadni eredményként.

A képen szöveg, étel, képernyőkép látható

Automatikusan generált leírás

*A model használata egyedi képpel és a kiírt eredmény*

## Használt technológiák

### Image 101 Dataset

Természetesen mivel képfelismerésről van szó, így a program egyik legfontosabb része maga a tanításhoz használt dataset. Az én választásom a Food 101 datasetre esett, ami Kaggle-ön elérhető bárki számára. Ez az adathalmaz 100.000 darab képet tartalmaz összesen 101 féle ételtípusról. A programomnak ebből 48 darab típus lett megtanítva a végső verzióban.

### PyTorch

A fejlesztés során a PyTorch gépi tanulási könyvtárra épült fel a programom. Azért választottam inkább ezt a TensorFlow-val szemben, mert képfelismeréshez sokkal jobban használható, több extension és library épült rá, amely ezzel foglalkozik (ilyen például a lentebb említett fast.ai).

### fast.ai

A fast.ai egy, a PyTorch-ra épülő gépi tanulási könyvtár, amelyet a gyors és hatékony mélytanulási modellfejlesztés támogatására terveztek. Ez a keretrendszer lehetővé teszi a gyors prototípuskészítést és a mélytanulási modellképzést minimális kódsorok felhasználásával.

### ResNet

A ResNet egy hatékony mély neurális hálózat architektúra. Előnyei közé tartozik a könnyebb tanulás és a mélyebb hálózatok építésének lehetősége.

A fast.ai használatával létrehozott modellem az előre tanított `resnet34` architektúrát alkalmazza, ami segít az általános képfelismerési feladatok hatékony megoldásában.

## Használati útmutató

### Github Repository

A projekt során használt GitHub repository linkje: <https://github.com/davidkalmn/image_classification_food_MI.git>

### Telepítési útmutató

1. Telepítsük a számítógépre a Python 3.11-es verzióját (a PyTorch Windowson csak a 3.8 és 3.11 közötti verziókat támogatja, így célszerű ezek közül a legfrissebbet használni a hibák elkerülése végett).
2. Klónozzuk a GitHub repository-t a kívánt célmappába terminálon keresztül:
   * *git clone https://github.com/davidkalmn/image\_classification\_food\_MI.git*
3. Telepítsük a szükséges csomagokat terminálon keresztül:
   * *pip install torch torchvision torchaudio*
   * *pip install -Uqq fastai*

### Használata fájlból

Ha lokálisan, fájlból szeretnénk használni a Foodifier-t:

1. A klónozott git mappán belül keressük meg a ***/model/use\_food\_model.ipynb*** fájlt, majd nyissuk meg Visual Studio Code-al, vagy valamilyen ehhez hasonló fejlesztői környezettel.
2. A megnyitott Jupiter Notebook kiterjesztésű fájlban futtassuk le az első 2 kódblokkot, amelyek az Előkészületekhez tartoznak.
3. Az utolsó kódblokkban, a 3. sorban adjuk meg a *local\_image\_path* változó értékeként a tesztelésre szánt kép elérési útvonalát, majd futtassuk le a kódblokkot.
4. A program végül megmutatja az általunk használt képet, valamint kiírja, hogy ez szerinte milyen típusú étel, illetve, hogy mekkora eséllyel „tippeli” a típusát.

## A program bemutatása

### Szerkezeti felépítés

A program szerkezetét 3 nagy csoportra lehet bontani:

* **Model**: 3 külön fájlban, a model létrehozása, további optimalizálása, valamint a model használata. Ez a három fájl a ***/model*** mappán belül található.
* **Assets**: Ide tartozik maga a dataset, amelyet a model létrehozásánál és optimalizálásánál használunk. Ezt automatikusan feltelepíti a program a gépünkre, amikor legelőször elindítjuk.
* **Test images**: ezek azok a képek, amelyeket nem a model training-elésére használunk, hanem a tesztelésére, hogy ténylegesen jól működik-e.

### A model elkészítése

A model elkészítése a ***/model/create\_food\_model.ipynb*** fájllal történik, azon belül is 4 lépésben:

1. **Load and prepare data**: az itt megtalálható kód fogja importálni a fast.ai-t a projektbe, hogy később használni tudjuk a parancsait, valamint ez telepíti fel a dataset-et is a gépre (végül egy tesztet is futtat, hogy sikeres volt-e a telepítés).

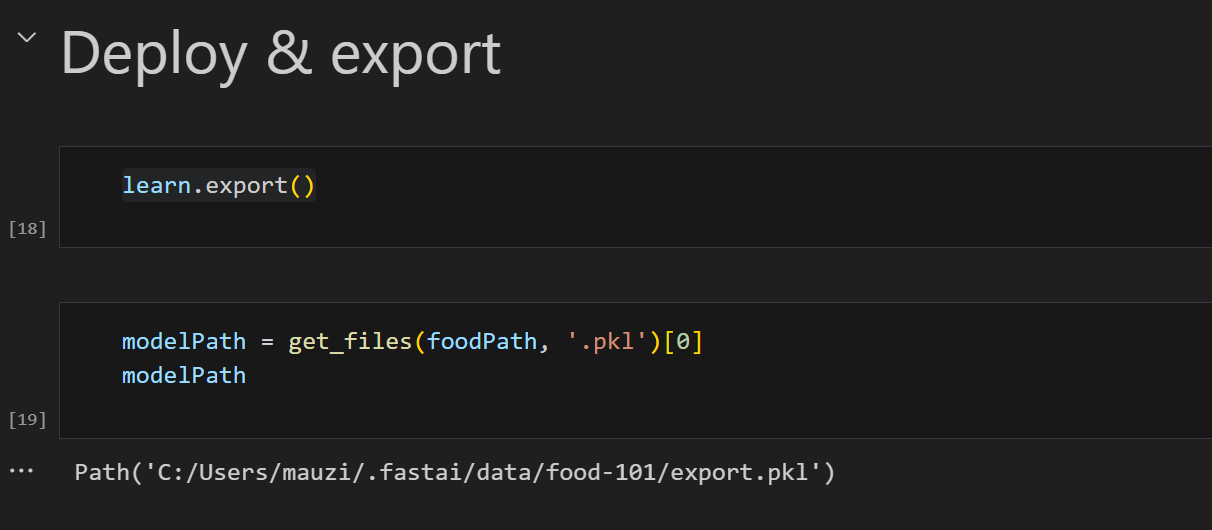
A képen szöveg, képernyőkép, szoftver, Multimédiás szoftver látható

Automatikusan generált leírás

1. **Training**: itt a program először létrehozza a label-öket (azaz az ételtípusokat), majd a fast.ai rövidített parancsai által létrehozza a model első változatát (jelen esetben a 48000 kép 80%-át használtam trainingre, a maradék 20%-ot tesztelésre). A fejlesztés során kipróbáltam több ResNet típust is (ResNet50 és ResNet100), minél magasabb a szintje, minél nagyobb a modelszám, annál pontosabb a végeredmény, viszont drasztikusan tovább is tart a létrehozás (már az 50-es verziónál is több mint 20 perc volt csak az első training 5-10 perc helyett).

A képen szöveg, képernyőkép, szoftver, Multimédiás szoftver látható

Automatikusan generált leírás

1. **Verify model**: ez a hitelesítési folyamat, itt használhatjuk a dataset maradék 20%-át, vagy pedig megadhatunk saját képeket is, ezzel is ellenőrizve, hogy biztosan jól működik-e a program.
2. **Deploy & export**: az utolsó lépésben exportáljuk a létrehozott és letesztelt modelt, hogy a továbbiakban másik fájlokban is tudjuk akár tovább finomítani vagy tesztelni.

### A model finomítása

A model finomítása a ***/model/train\_food\_model.ipynb*** fájlban történik viszonylag egyszerűen. Beimportáljuk a már meglévő, elmentett modelünket, betöltjük a kívánt taníttatási, finomítási metódust, megadjuk az epoch-ok számát (hogy hányszor, hány körben csinálja újra az optimalizálást, minél több, annál pontosabb lesz), majd elindítjuk a finomítást.

A képen szöveg, képernyőkép, szoftver, Multimédiás szoftver látható

Automatikusan generált leírásEz rengeteg időt vehet igénybe, amely függ a használt képek méretétől, mennyiségétől, valamint a hardver erősségétől. Az általam használt mód 5 ismétléses volt, ismétlésenként tartott kb. 18-19 percbe (összesen közel 1,5 óra). Ennek a lényege az, hogy minden ismétléssel, minden új finomítási folyamattal egyre pontosabb és pontosabb lesz a model (volt olyan teszt modellem, amely az eredeti 0.86-os hibarátáról ment le 0.36-ra közel fél napnyi folyamatos training után).

### A model használata

A model használata a ***/model/use\_food\_model.ipynb*** fájllal történik. 3 rövid kódblokkból áll, az első importálja a fast.ai-t, a következő létrehozza az ételkategóriákat, az utolsó pedig betölti a modelt, majd használja az általunk megadott képen. Eredményül látunk egy javaslatot, hogy a program szerint milyen étel van a képen, illetve egy hozzá tartozó valószínűséget.

## Irodalomjegyzék

**Dataset:**

* <https://www.kaggle.com/datasets/dansbecker/food-101>

**PyTorch:**

* <https://pytorch.org/tutorials/>
* <https://www.learnpytorch.io/>

**fast.ai:**

* <https://www.fast.ai/>
* <https://course.fast.ai/>

**Resnet:**

* <https://www.geeksforgeeks.org/residual-networks-resnet-deep-learning/>
* <https://medium.com/@ibtedaazeem/understanding-resnet-architecture-a-deep-dive-into-residual-neural-network-2c792e6537a9>
* <https://roboflow.com/model/resnet-34>