NEURAL NETWORKS FOR VISUAL CLASSIFICATION AND INSPECTION OF THE INDUSTRIAL PRODUCTS

Vojtěch Míček

Master Degree Programme , FEEC BUT

E-mail: xmicek04@stud.feec.vutbr.cz

Supervised by: Petr Petyovský

E-mail: petyovsky@feec.vutbr.cz

Abstract: The aim of this thesis is to enable evaluation of quality, or the type of product in industrial applications using artificial neural networks, especially in applications where the classical approach of machine vision is too complicated. The system thus designed is implemented onto a specific hardware platform and becomes a subject to the final optimalisation for the hardware platform for the best performance of the system.

Keywords: Neural networks, machine vision, product inspection, object classification, Nvidia Jetson Xavier, Raspberry Pi, Intel Movidius, OpenCV, Keras, TensorFlow

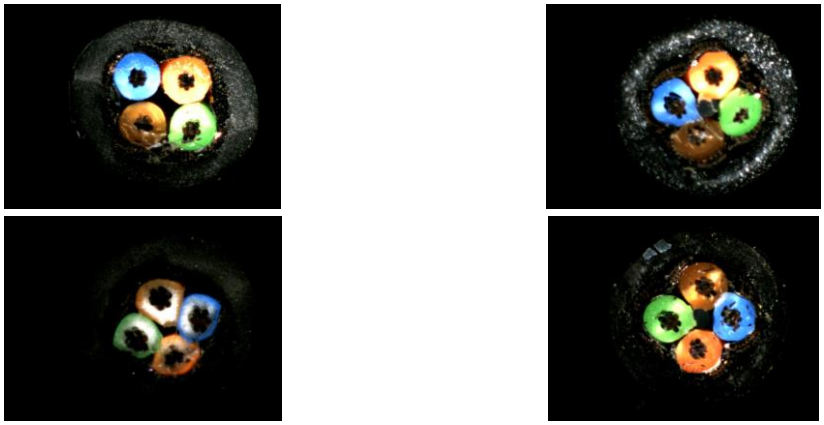
1 ÚVOD

Cílem práce je umožnit posuzování kvality nebo typu výrobku v průmyslových aplikacích pomocí umělých neuronových sítí. Takto navržený systém bude dále implementován na konkrétní hardwarovou platformu a bude provedena optimalizace výsledného modelu pro rychlejší běh systému. Díky navrženému systému, by tedy mělo být možné řešit vizuální strojovou kontrolu a kategorizaci výrobků i v některých případech, ve kterých to běžnými systémy doposud nešlo anebo by to bylo příliš náročné.

2 VÝBĚR VHODNÉ TŘÍDY VÝROBKŮ

Zadání diplomové práce vychází z požadavků společnosti TE Connectivity. Práce se tedy zaměřuje na stávající i budoucí projekty obrazové kontroly produktů této společnosti. Po prozkoumání vhodnosti byly vybrány 3 takové, které jsou běžnými způsoby velmi obtížně řešitelné. Dále bude představena jedna z nich.

2.1 ROZPOZNÁNÍ TYPU USTŘIŽENÉHO KABELU



Obrázek 1: 4 ze 6 typů ustřižených kabelů k rozpoznání

Jedná se o rozpoznání typu kabelu po strojovém ustřihnutí. Kabelů je 5 typů a v případě jednoho se ještě dělí na dvě kategorie podle použití typu střihačky.

3 TVORBA MNOŽINY TESTOVACÍCH SNÍMKŮ

Pro tvorbu množiny testovacích snímků bylo využito již zavedených kamerových systémů. Pro vybrané úlohy to jsou systémy založeny na systémech Cognex In-Sight Spreadsheet. Ve výsledku bylo zaznamenáno 1000 snímků od každé kategorie. Tedy v případě rozpoznávání typu ustřiženého kabelu to činí 6000 snímků. Na závěr byly snímky dotříděny manuálně. Vzhledem k velikosti tohoto datasetu a případné náročnosti zpracování snímků NN v plném rozlišení, byly tyto snímky upraveny na rozlišení 800 x 600 px.

4 VÝBĚR HARDWAROVÝCH PLATFOREM

4.1 HARDWAROVÁ PLATFORMA PRO UČENÍ

Po počátečním průzkumu trhu a zvážení lokálních i cloudových řešení, bylo rozhodnuto pro stavbu vlastní výpočetní stanice. Je založena na grafické kartě NVIDIA GeForce RTX 2080 Ti s 11 GB grafické paměti (bohužel pro některé modely nedostatečná hodnota), dále procesoru AMD Ryzen 9 3900X, 64 GB RAM (3600 MHz, CL16) a SSD ADATA XPG GAMMIX S50 1TB PCI-E 4 (z cenových důvodů nezvolen INTEL Optane).

4.2 HARDWAROVÁ PLATFORMA PRO PREDIKCI

Po důkladném zvážení dostupných možností byl zvolen NVIDIA Jetson AGX Xavier, jedná se o kompaktní (10,5 x 10,5 x 6,5 cm) jednodeskový počítač určený pro zpracování obrazu a inferenci neuronových sítí. Nabízí velké možnosti konektivity (PCI-E x16, 2x M.2 slot jeden z nich NVMe, HDMI 2.0, Gigabit Ethernet, USB 3.1 porty a CSI-2 Lanes). Jádrem zařízení je ARM v8.2 64-bit CPU, 8MB L2 + 4MB L3 a grafická karta Nvidia s architekturou Volta a Tensor akceleračními jádry. Byla přidána Wi-Fi karta Intel 7265NWG a SSD. V současnosti je navrhován držák zařízení, kompatibilní s DIN systémem, aby mohlo být zařízení umístěno do elektrického rozvaděče na výrobních linkách.

Obrázek 2: Hardwarová platforma NVIDIA Jetson AGX Xavier určena pro inferenci

5 IMPLEMENTACE APLIKACE PRO OPTICKOU KLASIFIKACI VÝROBKŮ

Nejdříve bylo experimentováno s vlastními architekturami konvolučních NN. Byla vyzkoušena klasickou úlohu rozpoznávání ručně psaných číslic s využitím volně dostupného MNIST datasetu. Následně pak rozpoznáváním rozpoznávání obličejů. Zde bylo rozhodnuto, že pro složitější klasifikace bude vhodnější využít již existujících architektur. Po několika pokusech byla zvolena architektura neuronové sítě Inception-v3. V následující části této práce bude vyzkoušena také nová verzi Inception-v4 a NASNet (pro vysokou výpočetní náročnost zatím vynechán).

6 INSTALACE SOFTWARU PRO UČENÍ A INFERENCI NN

V případě, že chceme využít pro učení modelu grafickou kartu NVIDIA z pracovní stanice (část 3.1.4), je zapotřebí nainstalovat značné množství podpůrného softwaru o přesně daných verzích. Tato oblast bohužel není velmi dobře dokumentovaná u oficiálních zdrojů a občas jsou tyto informace dokonce v rozporu s reálnou funkčností a tak je potřeba s verzemi experimentovat.

Aktuálně stabilní je následující kombinace:
- Keras 2.3.1
- TensorFlow 2.0
- Cuda 10
- CUDNN 7.6.0
- OpenCV 4.1.2.30

7 VÝSLEDKY KLASIFIKACE NATRÉNOVANOU NEURONOVOU SÍTÍ

Byla ověřena funkce neuronové sítě na úloze ustřiženého kabelu. Nejdříve byla natrénována neuronová síť s architekturou Inception-V3. Snímky byly rozděleny následovně. 700 od každé kategorie bylo určeno pro samotné trénování, 200 pro testování a 100 pro závěrečné, plně nezávislé ověření funkčnosti. Při trénování modelu bylo nastaveno 20 etap, což odpovídá přibližně 20 hodinám učení na dříve zmíněné pracovní stanici. Výsledek nezávislého testování, tedy inference naučeného modelu NN na snímcích, se kterými se neuronová síť nikdy nesetkala, je popsán v následující tabulce 1. Tabulka udává, že s výjimkou detekce manuálního střihu („man“) predikuje NN na ověřovacích snímcích jednotlivé typy kabelů dokonale. V případě „man“ se projevuje to, že se jedná o stejný typ kabelu jako je typ 5352, ale byl proveden jen jiný typ střihu. V tomto případě to nehraje pro výrobu žádnou roli a tuto nepřesnost tedy můžeme zanedbat, nicméně i tak se bude touto nepřesností do budoucna zabýváno.

	535 predikováno	5352 predikováno	blue predikováno	dacar predikováno	man predikováno
535 skutečné	100	0	0	0	0
5352 skutečné	0	100	0	0	0
blue skutečné	0	0	100	0	0
dacar skutečné	0	0	0	100	0
man skutečné	0	24	0	0	76

Tabulka 1: Tabulka výsledků predikce natrénované neuronové sítě Inception-v3 pro určení typu kabelu (tzv. confusion matrix, běžně používaná pro posuzování výsledků neuronových sítí)

8 OPTIMALIZACE NATRÉNOVANÉHO MODELU NEURONOVÉ SÍTĚ PRO NVIDIA JETSON XAVIER

8.1 TENSORRT

TensorRT je optimalizační knihovna pro NVIDIA grafické karty a NVIDIA zařízení z řady Jetson. Knihovna podporuje optimalizaci modelů, vytvořených ve většině frameworků pro neuronové sítě. Jsou podporovány TensorFlow, Caffe, PyTorch, MXNet a další. Základem této knihovny je to, že optimalizuje model neuronové sítě na konkrétní hardwarovou platformu. Optimalizované modely tedy nejsou přenosné mezi různými modely zařízení. Stejně tak jsou optimalizované modely vázá-

ny na konkrétní verzi Tensor RT. Samotná optimalizace probíhá tak, že se načte podporovaný model, odstraní se vrstvy a operace, které nemají žádný efekt na výsledek, následně spojí konvoluci, bias a ReLu operace do bloků. V závislosti na použité hardwarové platformě zvolí optimální algoritmy a datovou strukturu. Také minimalizuje potřebnou velikost paměti tím, že zřetězí výstupy z některých vrstev, a pokud používá stejná data více operací, zamezí jejich duplikaci. Dalším krokem je optimalizace změnou velikosti jednotlivých proměnných, ve většině případů se volí INT8. U žádného z modelů nebyla zaznamenána zhoršenou přesnost po provedení optimalizace. Optimalizace modelů, které jsou specifické pro konkrétní hardware, jsou prováděny tak, že knihovna vygeneruje testovací data a postupným spouštěním testuje na skutečném hardwaru, jaké nastavení a jaké optimalizace fungují nejlépe.

8.2 OPTIMALIZACE KERAS MODELU POMOCÍ TENSORRT

Vzhledem k tom, že nadstavbový framework Keras není knihovnou TensorRT podporovaný, je zapotřebí nejprve převést výsledný model do formátu, jaký by vytvořil TensorFlow. Již při samotném výpočtu Keras modelu je potřebné myslet na to, že bude model optimalizován. Je kritické, aby výstupní vrstva nebyla bez jména. Pokud bez jména je, nijak to neovlivňuje využití modelu k inferenci, ale zamezí tomu, aby mohl být model převeden do TensorFlow. Pokud je tedy připravený model, lze přistoupit k samotnému procesu optimalizace pomocí navrženého skriptu. Po spuštění vyzve ke zvolení cesty vstupního Keras modelu. Po načtení je model otevřen, jsou z něj vyčteny naučené váhy, architektura a název výstupní vrstvy a je uložen ve formátu tzv. frozen grafu TensorFlow. Tento frozen graf je následně načten a optimalizován knihovnou TensorRT a po dokončení optimalizace uložen. Rozsáhlejší možnosti a výkonové benefity takto provedené optimalizace budou předmětem testování v následující fázi této práce. V současném stavu jsou optimalizované modely o velikostech 200 MB – 700 MB na Nvidia Jetson Xavier, přibližně 6x rychlejší, než neoptimalizované modely spuštěné na profesionální grafické kartě NVIDIA Quadro M2200.

9 ZÁVĚR

Tato práce se zabývala klasifikací/detekcí vad výrobků pomocí neuronových sítí. Byly vybrány vhodné třídy úloh pro testování a zvoleny ty, které jsou standardními prostředky obrazové kontroly jen velmi obtížně realizovatelné. Byl vytvořen dataset snímků. Pro prezentovanou úlohu je to 6000 anotovaných snímků. Byla vybrána hardwarovou platformu pro učení, která je založena na NVIDIA RTX 2080 Ti a platforma NVIDIA Jetson Xavier pro rozpoznávání. Byla vyvinuta a implementována aplikace pro učení a rozpoznávání a nainstalován nezbytný software pro funkčnost hardwaru. Aplikace je založena na architektuře Inception-v3. Následně byla neuronová síť natrénována a ověřena klasifikací pro ustřižené kabely. Pro 5 kategorií klasifikuje aplikace bezchybně, v případě poslední kategorie, která se liší pouze typem střihu, což není pro výrobu nijak důležité, je správně klasifikováno 76 ze 100 snímků. Následovala optimalizace modelu s pomocí knihovny NVIDIA TensorRT, díky tomu se rozpoznávání zrychlilo přibližně 6x (srovníní Jetson Xavier optimalizováno, Quadro M2200 neoptimalizováno).

REFERENCE

[1] Keras Documentation:[online].[cit. 1. 12. 2019]. Dostupné z URL: <https://keras.io/documentation/>.

[2] TensorFlow API Documentation:[online].[cit. 1. 12. 2019]. Dostupné z URL: <https://www.tensorflow.org/api_docs>.

[3] TensorRT 7.0.0 Developer Guide:[online].[cit. 1. 12. 2019]. Dostupné Z URL: <https://docs.nvidia.com/deeplearning/sdk/tensorrt-developer-guide/index.html>.