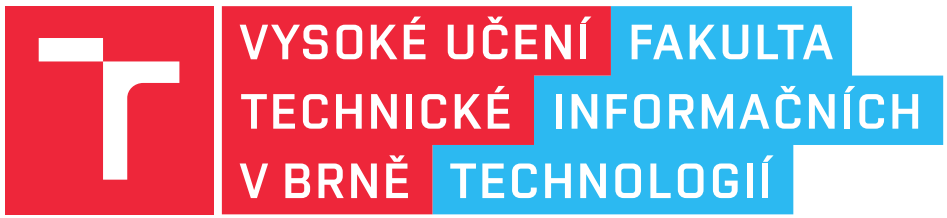
Generátory vah pro CNN

David Hudák

Fakulta informačních technologií Vysokého učení technického v Brně
Božetěchova 1/2. 612 66 Brno - Královo Pole

xhudak03@fit.vutbr.cz



30. dubna 2023

Zadání

Natrénujte menší CNN (např. LeNet) pro klasifikaci jednoduššího problému (např. MNIST, Cifar-10). Získanou přesnost klasifikace považujte za referenční. Rozdělte množinu vah na vhodné podmnožiny (např. dle vrstev apod.). Pro každou podmnožinu vah se pokuste najít matematický výraz, který ji co nejlépe aproximuje. Matematický výraz bude mít na vstupu vámi zvolené příznaky (odvozené z parametrů dané části sítě, popř. generátor čísel ve zvoleném rozsahu). Matematický výraz nalezněte pomocí genetického programování a symbolické regrese - můžete použít libovolný SW. Pokuste se minimalizovat složitost výrazů. Pomocí vytvořených výrazů vygenerujte váhy, které budou aproximovat původní váhy v dané podmnožině. Tyto nové váhy použijte namísto původních vah CNN a zjistěte, jak se změní přesnost klasifikace vzhledem k referenční hodnotě. Experimenty s přesností klasifikace proveďte pro všechny podmnožiny nezávisle.

Zadání (2)

Nakonec se pokuste nahradit co nejvíc podmnožin vah pomocí vytvořených generátorů. Proč se to celé dělá? V HW akcelerátoru CNN spotřebovává velké množství energie přesun vah z paměti. Zde je cílem určité podmnožiny vah nějak jednoduše generovat přímo na čipu.

- Zadání projektu – aproximovat váhy v konvoluční neuronové síti s pomocí symbolické regrese
- Pro symbolickou regresi využito kartézské genetické programování
- Použity různé strategie pro aproximaci vah
- Experimentováno primárně na vlastní neuronové síti se 3 konvolučními vrstvami a část experimentů na AlexNetu
-
  - U vlastní sítě: 1600 + 102400 + 9216 = 113216 vah konvoluce
  - AlexNet – Ze 62 milionů parametrů 40 960 vah poslední plně propojené vrstvy a 3 747 200 vah konvolučních jader
  - Sítě upraveny a trénovány na dataset MNIST (AlexNet – rozšíření MNIST obrázků a zmenšení výstupní vrstvy z 1000 na 10)
  - Experimentováno ještě se sítí LeNet – mnohem horší výsledky

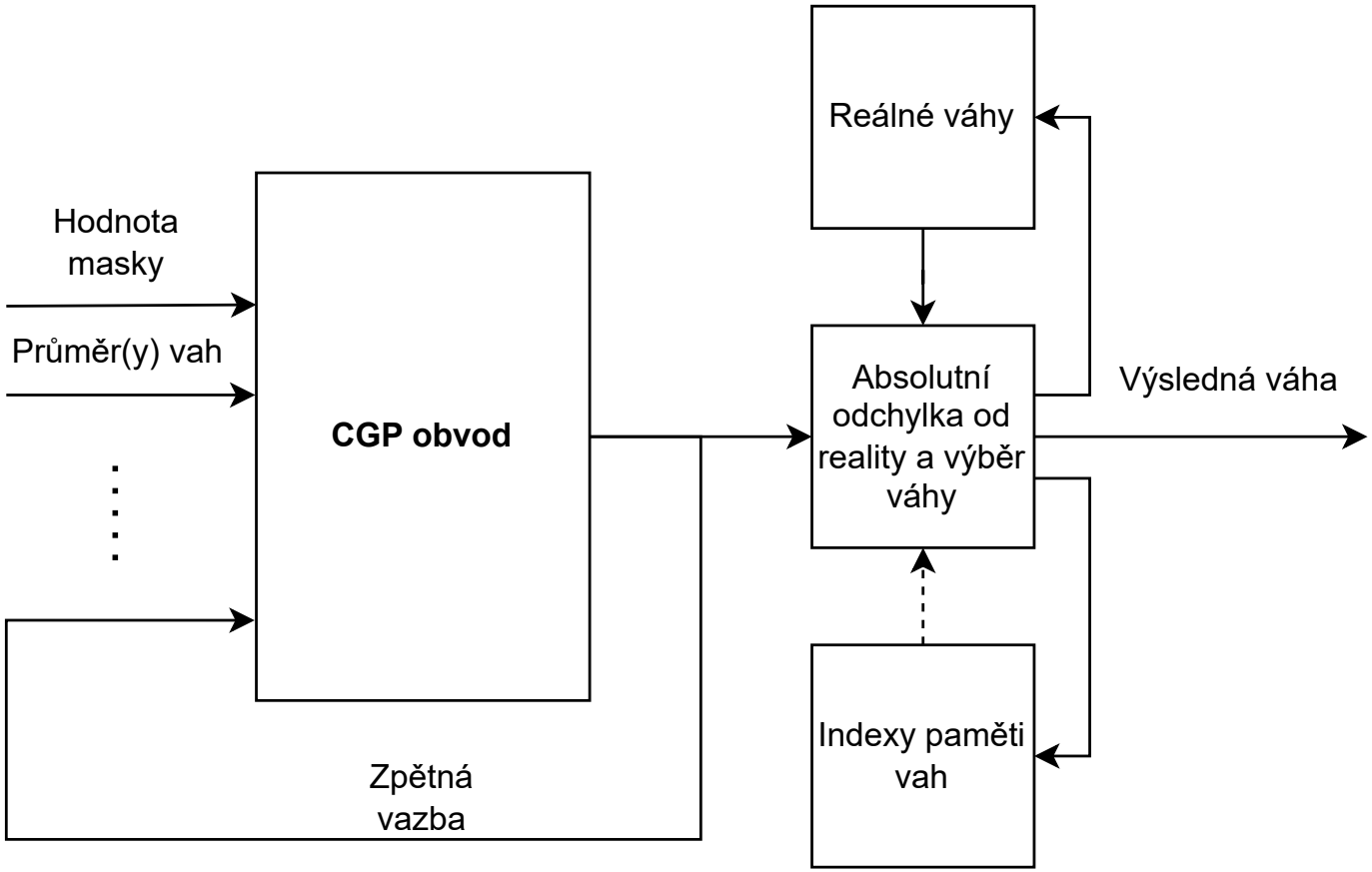
Vývojové prostředí

- Python s knihovnami Torch, Torchvision (modely), numpy…
- Použita knihovna HAL CGP – práce s desetinnými čísly
  - https://happy-algorithms-league.github.io/hal-cgp/
- Velikost obvodu 8×8
  - Větší obvod se při MNISTU neprokázal jako přínosný
- Od 5 do 8 vstupů obvodů (průměr, maska, kanál…), výchozí paměť 10 %
- Použito 100 generací pro každý generátor, na každý generátor 3 pokusy (vyjma AlexNetu)
- Fitness MSE reálných oproti generovaným váhám
- Referenční úspěšnost AlexNetu je 98.88 % na testovací sadě
- Referenční úspěšnost vlastní sítě je 99.02 %
- Celkem provedeno 105 běhů na menší síti a kolem 20 na AlexNetu

Použité strategie

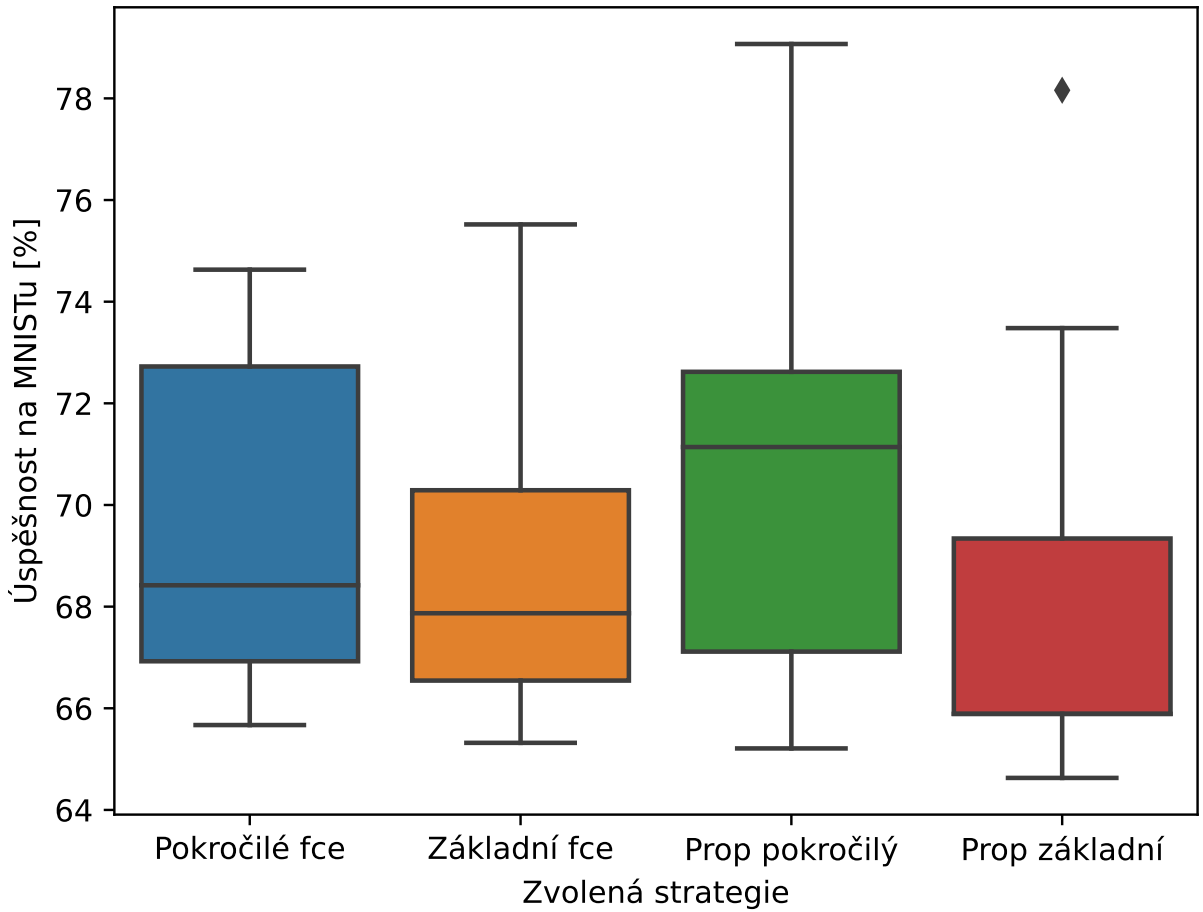
- Váhy v konvolučních vrstvách byly dělány stylem jedna vrstva, jeden obvod
- Váhy v plně propojených vrstvách dvojím způsobem
  1. Váhy jednoho výstupního neuronu, jeden obvod – pro velké množství vah vhodná varianta
  2. Všechny váhy najednou jedním obvodem – k ničemu
- Použity dva přístupy k blokům CGP
  1. Použity pouze stavební bloky ve výchozí knihovně (+ identita)
  2. Přidány bloky vlastní (sin, cos, log, exp, sqrt, avg, abssub)
- Použití zapínatelné zpětné vazby
- Pro konvoluční vrstvy použity hodnoty vstupního/výstupního kanálu, pozice v kanálu a průměr jádra
- Pro plně propojené vrstvy použity hodnoty průměru, rozptylu, maxima, minima a masky (normalizované hodnoty od 0 do počtu vah)

Obecné schéma obvodu



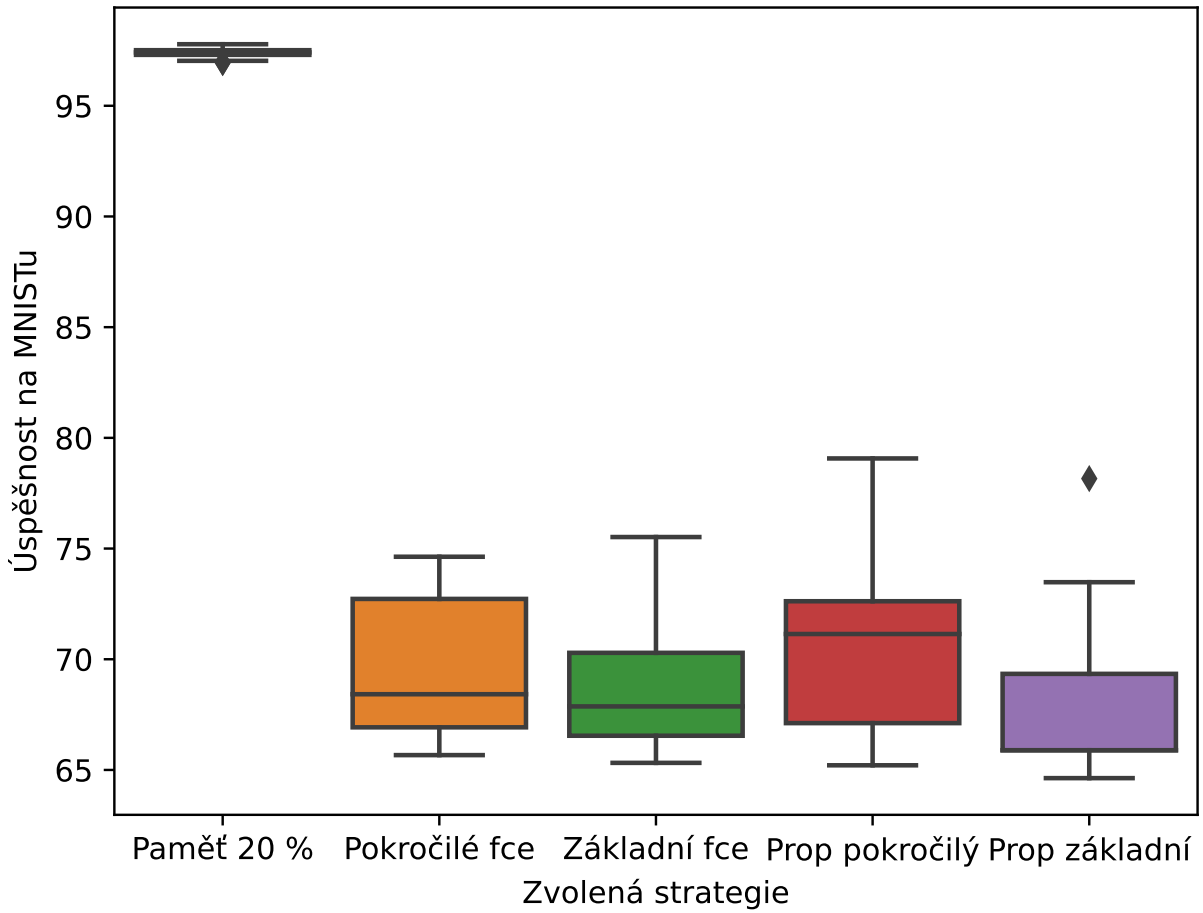
Obrázek: Zjednodušené schéma generování vah

Porovnání různých technik na menší síti



Obrázek: Jako nejlepší řešení se ukázal obvod se zpětnou propagací a pokročilými funkcemi

Ale. . .



Obrázek: Nic nepomůže tolik jako přidání více paměti.

Výsledky pro AlexNet

- Při 100 generacích a 15 bězích pro každou konvoluční a poslední plně propojenou vrstvu úspěšnost klasifikace „pouze“ 95.49 % (pád o 3.4 %)
- Pro konvoluční vrstvy pád jenom na 97.96 % (tedy necelé procento)
- V případě evoluce pouze poslední plně propojené vrstvy s 5 běhy na obvod, úspěšnost klasifikace padá jen na 98.21 % (z 98.88 %)
- Problém – pokud selže jeden obvod pro plně propojenou vrstvu, úspěšnost klasifikace vždy výrazně klesne
- Pokus o řešení – generovat obvody pro jednu konvoluční vrstvu/část plně propojené vždy několikrát a vybrat ten, který nejméně sníží úspěšnost klasifikace
  - Nejlepší řešení se povedlo vytvořit při mírném zvýšení paměti na 11, dosáhlo 97.81% úspěšnosti na testovacím datasetu

Shrnutí výsledků a potenciální vylepšení

- Zpětná vazba mnohdy vede k problémům se stabilitou řešení (přetečení hodnot)
- Lepší aproximace vah neznamená vždy lepší fungování sítě – vyplatí se testy sítě samotné
- Ztráta na klasifikační přesnosti kolem 3 % při 4 až 6-násobném snížení spotřeby paměti na 5 konvolučních a 1 plně propojené vrstvě
  - Většina ztráty je na plně propojené vrstvě! Konvoluční u AlexNetu se vejdou pod 1 %
- Finální výsledek je silně ovlivněn potenciálním selháním byť jediného obvodu
- Obecně čím větší vrstva, tím lepší aproximace (větší tolerance vůči chybě)
- Hodně místa zabírá paměť průměrů konvolučních jader a indexy, velikost obvodu pak bývá prakticky zanedbatelná oproti reálným vahám (klidně i 9000krát menší než reálné váhy)

Potenciální vylepšení a možné problémy

- Přidání paměti přímo na vstup obvodu
  - Například umožnit zpětnou propagaci a každý desátý prvek tahat z paměti
- Brát více kanálů najednou, ale ne všechny
  - Například jeden obvod pro kovoluční jádra pro jeden vstupní/výstupní kanál
- Problematická regrese plně propojených vrstev, kde je i nejvíce vah
  - Volba jiných funkcí, více generací
  - Seskupování podobných vrstev
  - Lepší práce s pamětí
- Více výpočetního času
- Problémy – jedná se o jednoduchý problém na velkých sítích, pro praktické využití nutné více experimentů
- Problémy – s pamětí se aktuálně ukládají i indexy proměnných a průměry konvolučních jader (řešení – asociativní výběr, výběr obvodem)

Díky za pozornost !