PREGLED PROGRAMSKIH ORODIJ ZA GLOBOKO UČENJE Z VIDIKA UPORABE V INDUSTRIJSKIH APLIKACIJAH

Domen Tabernik, Danijel Skočaj

Univerza v Ljubljani
Fakulteta za računalništvo in informatiko
E-pošta: domen.tabernik@fri.uni-lj.si
URL: http://www.vicos.si

POVZETEK: *Globoko učenje je prineslo revolucionarne spremembe na področju računalniškega vida in si utira svojo pot tudi na področje industrijskega strojnega vida. V tem članku predstavljamo šest najbolj poznanih orodij za delo z globokimi arhitekturami: Caffe, Torch, Theano, MatConvNet, TensorFlow in Keras. Predstavili bomo njihove glavne značilnosti tako z vidika razvoja kot integracije v industrijske aplikacije.*

1. UVOD

V zadnjih nekaj letih smo bili na področju računalniškega vida priča veliki spremembi v paradigmi reševanja problemov. S povečanjem računskih zmogljivosti in razpoložljivosti ogromnega števila slik ter z napredkom pri razvoju pristopov, ki temeljijo na principu globokega učenja, so se globoke arhitekture uveljavile kot najuspešnejša predstavitev vizualne informacije, ki omogoča uspešno reševanje različnih problemov kot so detekcija, razpoznavanje in kategorizacija predmetov in njihovih delov, ali semantična segmentacija.

Industrija je tradicionalno previdnejša pri sprejemanju novih tehnologij. Kljub temu je opazen trend prodiranja globokega učenja tudi na področje industrijskih aplikacij kot je nadzor kvalitete izdelkov. Avtomatsko učenje značilnic in klasifikatorjev na podlagi učnih primerov se namreč zdi v veliko primerih primernejše kot klasično programiranje namenskih detektorjev za vsak problem posebej. Zato lahko pričakujemo, da bo vse več rešitev na področju industrijskega strojnega vida temeljilo na globokih arhitekturah. Pri tem se postavlja vprašanje katera orodja za delo z globokimi arhitekturami so najprimernejša za razvoj takšnih rešitev in integracijo le-teh v sisteme strojnega vida.

V tem članku bomo predstavili pregled šestih najpomembnejših obstoječih programskih orodij za globoko učenje ter opisali njihove prednosti in slabosti. Orodja bomo analizirali predvsem z vidika njihove uporabnosti v aplikacijah strojnega vida kot so nadzor kvalitete, detekcija napak oz. klasifikacija površin industrijskih izdelkov in polizdelkov ipd. Povzeli bomo njihove lastnosti v fazi razvoja aplikacij kot tudi primernost za integracijo razvitih rešitev v celovite sisteme strojnega vida.

2. ORODJA

V tem poglavju so opisana sledeča orodja za globoko učenje: Caffe, Torch, Theano, MatConvNet, TensorFlow in Keras. Tabela 1 povzema nekatere glavne značilnosti teh orodij: katere jezike in operacijske sisteme podpirajo ter pod kakšno licenco so izdani. V nadaljevanju poglavja so ta orodja predstavljena bolj podrobno.

Tabela 1: Povzetek glavnih značilnosti šestih orodij za globoko učenje.

Orodje	Programski jeziki	Operacijski sistem	Licenca
Caffe [1]	C/C++ in CUDA jedro, C++, Python, Matlab vmesnik	Primarno Linux, Windows slabše podprt	BSD 2-Clause
Torch [2]	C/C++ in CUDA jedro, Lua vmesnik, Python	Primarno Linux, Windows slabše podprt	BSD 3-Clause
Theano [3]	Python, C in CUDA jedro	Primarno Linux, Windows slabše podprt	BSD
MatConvNet [4]	Matlab z Mex/C oz. CUDA jedri	Linux, Windows	Permissive BSD
TensorFlow [5]	C/C++ in CUDA jedro, Python/C++ vmesnik	Primarno Linux, MacOS, Windows	Apache 2.0
Keras [6]	Python vmesnik nad orodjem Theano ali TensorFlow	Odvisno od izbire jedra (Theano ali TensorFlow)	MIT

2.2 Caffe

Orodje Caffe [1] je osredotočeno na globoko učenje aplicirano na slikovne zbirke. V ta namen ima odlično podporo za modele, ki izhajajo iz konvolucijskih mrež (ConvNet) in zanje vsebuje ustrezne najnovejše module oz. nivoje kot so BatchNorm, Dropout, ReLU, PReLU itd. V zadnjem času se izboljšuje tudi podpora za ne-konvolucijske mreže, kot so rekurzivne mreže (RNN) z nivoji LSTM.

Jedro orodja Caffe se razvija v programskem jeziku C/C++, pri tem pa ima večina nivojev implementacijo tudi v programskem jeziku CUDA, kar omogoča izvajanje na grafičnih karticah na sistemih z enim in večimi grafičnimi procesorji. Večina nivojev ima implementacijo CUDA realizirano tudi s pomočjo knjižnice CuDNN, kar omogoča optimalno izvajanje mrež na grafičnih karticah nNvidia. Vmesnik do jedra je implementiran v jezikih C++, Python in Matlab. Uporaba vmesnika Python in Matlab omogoča hiter in prototipen razvoj metod, vendar le z uporabo že pripravljenih nivojev. Nove nivoje je sicer mogoče implementirati v jeziku Python ali Matlab, vendar je zaradi učinkovitega izvajanja na CPU in GPU potrebno nove nivoje implemetirati ročno v jezikih C/C++ in CUDA. Orodje Caffe tudi nima neposrednih orodij za vizualizacijo, vendar je to mogoče nadomestiti z vizualizacijskimi orodji iz knjižnic programskih jezikov Python in Matlab.

Glavni nosilec razvoja orodja Caffe je skupina Berkeley Artificial Intelligence Research (BAIR), obstaja pa tudi širša skupnost raziskovalcev, ki pomaga pri razvoju. Orodje Caffe je tudi eno izmed glavnih orodij raziskovalcev pri razvoju novih metod globokega učenja, zaradi česar je podpora za vzdrževanje in razvoj stabilna. Kot glavno orodje

raziskovalcev ima prednost v hitrem dostopu do najnovejših, najaktualnejših metod in ogromnem naboru pred-naučenih modelov, kot so AlexNet, VGG, ResNet, GoogLeNet in različnih pristopov kot so R-CNN, CPM, DCNN itd. Velika večina teh modelov je zbranih na uradni strani pod imenom Caffe Model Zoo [7].

Z vidika integracije ima Caffe dokaj enostaven proces prevajanja z uporabo orodja CMake. Pri tem ima sicer nekaj pomembnih odvisnosti od tretjih knjižnic: HDF5, protobuf, glog, gflags, hdf5, ATLAS/OpenBLAS, Boost in CUDA. Ker je Caffe primarno razvit za operacijski sistem Linux je proces prevajanja na tem sistemu dokaj enostaven, medtem ko za operacijski sistem Windows obstaja podpora le iz ločene veje GIT in ne zagotavlja najnovejših popravkov in nadgradenj.

Glavni problem orodja Caffe z vidika integracije je uporaba "globalnih" knjižnic kot so glog in gflags, ki skrbita za napake in jih vračata kot izhod iz programa. V primeru uporabe orodja Caffe kot ločene knjižnice to predstavlja velik problem. Odprava te napake bi zahtevala premik velikega dela kode, kar bi povzorčilo težave pri vzdrževanju kode ob dodajanju najnovejših popravkov.

2.3 Torch

Programsko orodje Torch [2] je splošno namenska knjižnica tako za globoko učenje, kot tudi za strojno učenje. Poglavitna lastnost orodja je v implementaciji globokega učenja z osnovnimi tenzorskimi operacijami, le-te pa so učinkovito implementirane bodisi v programskem jeziku C, bodisi v jeziku CUDA/CuDNN ali OpenCL za izvajanje na grafičnih karticah. Preko vmesnika je nato mogoče implementirati oz. uporabiti različne tipe globokega učenja, od konvolucijskih mrež (ConvNet) do rekurzivnih mrež (RNN) in modelov LSTM, omogoča tudi podporo za modele kot so variacijski avtoenkoderji in modelov RBM (ang. Restricted Boltzman Machine). Orodje ima tudi velik nabor splošno namenskih metod strojnega učenja, kot so K-Means, SVM, PCA, LDA itd. Orodje vsebuje veliko različnih najaktualnejših modelov in pristopov, vendar pa je z vidika konvolucijskih mrež teh modelov malenkost manj kot v primerjavi z orodjem Caffe. Obstaja tudi veliko pred-naučenih modelov, obenem pa je mogoče z uporabo neuradnih pretvornikov v Torch iz orodja Caffe pretvoriti večino modelov, ki uporabljajo le standarne nivoje.

Primarni vmesnik do orodja Torch je implementiran v programskem jeziku Lua, obstaja pa tudi poenostavljena podpora za MATLAB. Omeniti velja tudi PyTorch, ki se v zadnjem času aktivno razvija in omogoča vmesnik do orodja Torch preko programskega jezika Python. Hiter, prototipen razvoj je z jezikom Lua dokaj enostaven. Vmesnik Lua vsebuje tudi knjižnice za vizualizacijo preko ogrodja Qt ali knjižnice GNUPlot, pri čemer vmesnik priskrbi knjižnice za izris, vizualizacijo modela pa je podobno kot pri Caffe potrebno narediti ročno.

Torch je glavno orodje za raziskovalce iz skupine Facebook-AI (FAIR), zato je podpora pri vzdrževanju in razvoju kode zelo dobra, obenem pa obstaja tudi širša skupnost raziskovalcev, ki pri tem pomaga. Primarno se orodje Torch razvija za operacijski sistem Linux, medtem ko obstaja za operacijski sistem Windwos neuradna podpora s strani skupnosti. Z vidika integracije je Torch zelo kompleksno orodje z veliko paketi zato ima

posledično veliko število odvisnosti do tretjih knjižnic. Že osnovni paket vsebuje nekaj pomembnih odvisnosti (OpenBLAS, libjpeg, imagemagick, zeromq, graphicsmagick, xquartz), pri tem pa se za proces prevajanja ne uporablja avtomatskih orodij ampak običajne skripte. Za uporabo v tretjih aplikacijah predstavlja težavo integracija velikega nabora paketov, prav tako pa je prilagajanje procesa prevajanja lahko kompleksno opravilo.

2.4 Theano

Theano [3] je splošno namenska knjižnica za simbolično reševanje matematičnih optimizacijskih problemov. Pri tem je knjižnica idealna za probleme modelirane z računskim grafom. Glavna lastnost je prevajanje iz simbolične definicije problema neposredno v kodo C, kar omogoča generiranje močno optimizirane kode. Obenem omogoča prevajanje neposredno v kodo GPU (tudi CuDNN), s čimer postane knjižnica primerna za obdelavo ogromnih količin podatkov iz globokih mrež. Model računskega grafa je tudi primeren za implementacijo poljubnih nevronskih mrež, vključno s ConvNet, še bolj primeren pa je za rekurzivne mreže (RNN). Kot splošno namenska knjižnica Theano nima eksplicitno pripravljenih nivojev/mrež za globoko učenje in jih je potrebno ročno ustvariti s pomočjo vmesnika. Prav tako ne obstaja množica obstoječih pred-naučenih modelov kot pri orodjih Caffe in Torch. Obstaja pa neuradno orodje za pretvarjanje modelov iz orodja Caffe v orodje Theano s čimer je mogoče uporabiti nekatere pred-naučene modele iz orodja Caffe.

Theano vsebuje vmesnik implementiran v programskem jeziku Python, ki omogoča hiter in prototipen razvoj. Podobno kot orodje Caffe ne vsebuje knjižnic za vizualizacijo zato je potrebno uporabiti tretje knjižnice, kot so matplotlib. Razvoj orodja Theano podpira Montreal Institute for Learning Algorithms z Univerze v Montrealu.

Theano se razvija primarno za operacijski sistem Linux, obstaja pa tudi neuradna podpora za operacijsk sistem Windows, ki pa je lahko manj stabilna. Z vidika integracije orodje Theano ne vsebuje veliko odvisnih knjižnic. Glavne knjižnice so BLAS in pa Python knjižnice (NumPy, SciPy), kar močno poenostavi proces prevajanja. Glavni problem je vprašljiva modularnost, saj je API močno integriran v Python. Prav tako Theano nima eksplicitnega aplikacijsko programskega vmesnika za C/C++, kar onemogoča uporabo pred-naučenih modelov v produkcijskem okolju preko C/C++ vmesnika.

2.5 MatConvNet

MatConvNet [4] je implementacija konvolucijskih mrež (ConvNet) kot programski paket (ang. toolbox) za programski jezik MATLAB. Konvolucijske mreže so implementirane v kodi C in CUDA (tudi z uporabo knjižnice CuDNN) in imajo le vmesnik MEX za MATLAB. Orodje podpira le konvolucijske in enostavne polno povezane mreže. Pri tem ima podporo za nekatere najbolj popularne nivoje, kot so BatchNorm, ReLU, Dropout in ROIPooling, medtem ko je podpora za ostale tipe mrež, kot so rekurzivne mreže, pomanjkljiva. Orodje podpira večino popularnih pred-naučenih

mrež, kot so AlexNet, VGG, GoogLeNet in ResNet, manjka pa podpora za ostale najaktualnejše modele kot so semantična segmentacija itd. MatConvNet ne vsebuje orodij za vizualizacijo, vendar je preko vmesnika mogoče vizualizacijo narediti z različnimi funkcijami v MATLABu. Prav tako je mogoče hiter in prototipen razvoj s pomočjo okolja MATLAB.

MatConvNet je podprt s strani skupine z Univerze v Oxfordu pod vodstvom Andrea Vedaldija. Pri tem skupina sčasoma vključuje bolj pomembne modele in nivoje globokega učenja, vendar zaradi obširnosti področja lahko pokriva le najbolj razširjene modele.

Proces prevajanja je pri MatConvNet relativno preprost, vendar ne omogoča enostavne integracija v tretje aplikacije, saj je vmesnik omejen le na okolje MATLAB. Celotna koda je tudi zgrajena na uporabi vmesnika MEX, zato je praktična uporaba zunaj okolja MATLAB nemogoča.

2.6 TensorFlow

TensorFlow [5] je podobno kot Theano orodje za reševanje matematičnih problemov definiranih z računskimi grafi. Podobno kot Theano lahko orodje samodejno izračuna odvode in gradiente na podlagi matematične definicije s čimer omogoča enostavno in hitro prototipno razvijanje novih metod. Za razliko od orodja Theano ima pripravljenih večino pomembnih nivojev za globoko učenje (konvolucijski nivoji, max pooling, ReLU, dropout, BatchNorm, itd). Ker je osnovan na računskih grafih, ima tudi dobro podporo za rekurzivne mreže (RNN). Poleg osnovnih funkcij ima neuradno podporo za še večji nabor metod globokega učenja, kot so metode Monte Carlo, variacijske metode, CRF, itd. Za orodje TensorFlow obstaja nekaj pripravljenih modelov, kot so ResNet, Inception, avtoenkoderji, itd. Nekaj najbolj popularnih mrež je tudi pred-naučenih in pripravljenih za uporabo (VGG, ResNet), vendar je večina teh mrež razdrobljenih po različnih spletnih straneh in niso enostavno zbrani na enem mestu kot na Caffe Model Zoo. Obstaja tudi neuradno orodje za pretvarjanje modelov s standarnimi nivoji iz orodja Caffe v model za orodje TensorFlow. Ker raziskovalci ne uporabljajo pogosto orodja TensorFlow kot primarno orodje za razvoj, večina state-of-the-art modelov pride na orodje TensorFlow z manjšo zamudo.

TensorFlow ima jedro napisano v C/C++, ter v CUDA (tudi s knjižnico CuDNN) za GPU, medtem ko ima vmesnik implementiran za programska jezika Python in C++. Zaradi enostavne uporabe vmesnika Python omogoča hiter in enostaven prototipni razvoj metod. H temu pripomore tudi že obstoječa knjižnica za vizualizacijo (TensorBoard), ki omogoča hiter pregled nad delovanjem mreže.

Poglavitna prednost je tudi v odlični podpori za porazdeljeno procesiranje. Orodje TensorFlow podpira tako porazdelitev na več enot GPU znotraj enega računalnika, kot tudi enot porazdeljenih po več računalnikih. Pri tem se lahko računski model razdeli vertikalno (vsak nivo svoja naprava) ali horizontalno (vsaka naprava odgovorna za določen paket primerov). Slabost orodja TensorFlow je predvsem v zapletenosti

uporabe, ter v počasni implementaciji računskega grafa kadar je implementiran le kot model v jeziku Python.

Orodje TensorFlow uradno podpira korporacija Google zato ima zelo dobro urejeno podporo, v zadnjem času pa se izboljšuje tudi dokumentacija, ki je bila v preteklosti pomanjkljiva. Z vidika integracije ima TensorFlow primaren proces prevajanja preko orodja Python PIP, kjer pa lahko pride do konflikta pri odvisnosti od ostalih Pythonovih knjižnic, ki jih TensorFlow pripelje zraven. Temu se je mogoče izogniti z neodvisno namestitvijo, bodisi v ločene datatoke (virtualenv), bodisi z slikami Docker. V osnovi orodje TensorFlow zahteva le manjše število odvisnih tretjih knjižnic: Python, six, NumPy, SciPy, protobuf, werkzeug, wheel, Eigen (interno prevajanje). Omogoča pa proces prevajanja preko različnih orodij za avtomatizirano prevajanje, uradno preko orodij Python PIP, bazel in conda, in neuradno preko CMake. Uradna podpora obstaja za operacijske sisteme Linux, MacOS in Windows, pri čemer je za sistem Windows zahtevan Python 3.5. Obsežen postopek prevajanja lahko zaplete integracijo v tretje aplikacije, vendar če se integrirata le C++ vmesnik in jedro, je to mogoče narediti enostavno preko orodja bazel, ki prevede TensorFlow v deljeno knjižnico (ang. shared library), odpade pa tudi večina odvisnosti od knjižnic Pythona.

2.7 Keras

V nasprotju z ostalimi orodji je Keras [6] le programski ovoj nad obstoječim orodjem, ki podpira tenzorske operacije preko računskega grafa. Keras omogoča uporabo orodja Theano ali TensorFlow kot glavnega jedra, pri tem pa z enostavnejšim vmesnikom Python močno poenostavi razvoj. Keras ima pripravljene obstoječe nivoje, kot so konvolucijski nivo, polno povezani nivo, rekurzivni nivo, nivoji batch normalization, ReLU, PReLU, itd. Pri tem omogoča zelo enostavno združevanje nivojev za uporabo v globokem učenju in je primeren za uporabo pri navadnih konvolucijskih mrežah kot tudi rekurzivnih mrežah. Keras omogoča tudi enostavno vizualizacijo, poleg tega pa omogoča izpis podatkov v obliki primerni za TensorBoard, s čimer se odpre vizualizacija z orodji TensorFlow.

Orodje podpira tudi nekaj popularnih že nučenih modelov, kot so VGG, ResNet in Inception, obstajajo pa tudi neuradna orodja, s katerimi je mogoče pretvoriti modele iz orodja Caffe v modele primerne za orodje Keras. Glede na dokumentacijo pa ne obstaja uradnega postopka za uporabo pred-naučenih modelov iz orodja TensorFlow neposredno v orodju Keras.

Ker predstavlja Keras le dodaten vmesnik v jeziku Python ni primeren za integracijo v tretje aplikacije. Vendar, če se uporablja orodje TensorFlow, je možno modele narejene v okolju Keras izvoziti neposredno v modele za orodje TensorFlow, s čimer je mogoče model uporabiti v vmesniku C++ do orodja TensorFlow.

3. ZAKLJUČEK

Izbira ustreznega orodja bo predvsem odvisna od specifičnih zahtev bodisi z raziskovalnega bodisi z integracijskega vidika industrijskih aplikacij. V primeru le raziskovalne uporabe je zelo primerno orodje MatConvNet, saj omogoča hiter proces prevajanja in enostavno uporabo v programskem okolju MATLAB. Slabost je v zelo omejenem dostopu do najaktualnejših modelov in pristopov, ter v nezmožnosti integracije v industrijske aplikacije. Podoben problem velja tudi za orodje Theano, ki ima sicer zaradi splošne namenskosti večjo uporabnost, vendar ima zelo omejen nabor pred-naučenih modelov in nezmožnosti integracije v industrijske aplikacije.

V primeru zahteve po uporabi najnovejših pristopov so primerna orodje Caffe, Torch in TensorFlow, pri čemer ima orodje Caffe največji nabor najnovejših modelov za konvolucijske mreže, saj večina raziskovalcev razvija prav v tem orodju. Slabost orodja Caffe je v omejenosti na konvolucijske nivoje, v zahtevi po ročni implementaciji novih nivojev v jezikih C++ in CUDA, ter v pomanjkljivi modularni zasnovi, ki otežuje integracijo. V primeru, da so najaktualnejši modeli narejeni s standardnimi nivoji, je vse modele mogoče z manjšim naporom pretvoriti iz modelov orodja Caffe v modele za orodje Torch ali TensorFlow, zato sta tudi ti dve orodji primerni z raziskovalnega vidika. Glavna prednost orodja Torch je ogromna zbirka funkcionalnosti, ki se razteza od običajnih globokih mrež do generativnih mrež (ang. variational autoencoders), in do splošnih funkcij strojnega učenja (SVM, PCA, LDA, itd.). Slabost orodja Torch je predvsem kompleksen postopek integracije v industrijske aplikacije, ter tudi manj znani jezik vmesnika (Lua).

V nasprotju z orodji Caffe in Torch ima TensorFlow najmanjšo podporo za prednaučene najaktualnejše modele, vendar, ker obstajajo orodja za pretvarjanje modelov iz orodja Caffe v modele za orodje TensorFlow, ostane glavna slabost le pri modelih, ki ne uporabljajo standardnih nivojev kot so npr, Faster R-CNN. Poglavitna prednost orodja TensorFlow je v že-vgrajenih vizualizacijskih orodjih, v dokaj veliki zbirki pomembnih nivojev globokega učenja ter v enostavni uporabi vmesnika Python tudi s knjižnico Keras. Prav tako omogoča orodje TensorFlow enostavno uporabo modelov preko vmesnika C++, ima podporo na različnih strojnih arhitekturah (Android, GPU, CPU itd), ter edini omogoča porazdeljeno učenje preko različnih računalnikov in različnih grafičnih kartic. Tudi z vidika integracije v ločene aplikacije ima orodje TensorFlow še najboljšo podporo za integracijo z neposrednim vmesnikom preko jezika C++ in možnostjo prevajanja v ločeno knjižnico, kar močno olajša integracijo v aplikacije in kasnejše vzdrževanje aplikacije.

LITERATURA

1. Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, T. Darrell, "Caffe: Convolutional Architecture for Fast Feature Embedding", *arXiv preprint arXiv:1408.5093*, 2014, Online: http://caffe.berkeleyvision.org

2. Torch, a scientific computing framework for LuaJIT, Online: http://torch.ch

3. Theano Development Team, "Theano: A Python framework for fast computation of mathematical expressions", *arXiv e-prints, abs/1605.02688*, 2016, Online: http://arxiv.org/abs/1605.02688, Online: http://deeplearning.net/software/theano/

4. A. Vedaldi and K. Lenc, "MatConvNet - Convolutional Neural Networks for MATLAB", In *Proceeding of the ACM Int. Conf. on Multimedia*, 2015, Online: *http://www.vlfeat.org/matconvnet/*

5. Tang, Yuan. "TF.Learn: TensorFlow's High-level Module for Distributed Machine Learning." *arXiv preprint arXiv:1612.04251* (2016), Online: https://www.tensorflow.org

6. F. Chollet, "Keras.", In *GitHub*, 2015, Online: https://github.com/fchollet/keras

7. Caffe Model Zoo, Online: https://github.com/BVLC/caffe/wiki/Model-Zoo